Dental Clinic Management System Requirements Specification

Version 1.0

April 2, 2018

*To regenerate the TOC, select all (CTL-A) and press F9.*

**Table of Contents**

[1.Executive Summary 3](#_Toc510459586)

[1.1 Project Overview 3](#_Toc510459587)

[1.2 Purpose and Scope of this Specification 3](#_Toc510459588)

[2. Product/Service Description 4](#_Toc510459589)

[2.1 Product Context 4](#_Toc510459590)

[2.2 User Characteristics 4](#_Toc510459591)

[2.3 Assumptions 4](#_Toc510459592)

[2.4 Constraints 4](#_Toc510459593)

[2.5 Dependencies 5](#_Toc510459594)

[3. Requirements 5](#_Toc510459595)

[3.1 Functional Requirements 5](#_Toc510459596)

[3.2 Non-Functional Requirements 7](#_Toc510459597)

[3.2.1 User Interface Requirements 7](#_Toc510459598)

[3.2.2 Usability 7](#_Toc510459599)

[3.2.3 Performance 7](#_Toc510459600)

[Capacity 7](#_Toc510459601)

[Availability 8](#_Toc510459602)

[Latency 8](#_Toc510459603)

[3.2.4 Manageability/Maintainability 8](#_Toc510459604)

[3.2.4.1 Monitoring 8](#_Toc510459605)

[3.2.4.2 Maintenance 8](#_Toc510459606)

[3.2.4.3 Operations 8](#_Toc510459607)

[3.2.5 System Interface/Integration 8](#_Toc510459608)

[Network and Hardware Interfaces 9](#_Toc510459609)

[Systems Interfaces 9](#_Toc510459610)

[3.2.6 Security 9](#_Toc510459611)

[Protection 9](#_Toc510459612)

[Authorization and Authentication 9](#_Toc510459613)

[3.2.7 Data Management 9](#_Toc510459614)

[3.2.8 Standards Compliance 10](#_Toc510459615)

[3.2.9 Portability 10](#_Toc510459616)

[3.3 Domain Requirements 10](#_Toc510459617)

[4. User Scenarios/Use Cases 10](#_Toc510459618)

# 1.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 | The system should handle multiple account types. | Based on the status of the user, each will have their own view of the system | 1 |  | Bob Dylan, Mick Jagger |
| R\_D\_1 | Keeping track of medical records and past treatments history | Doctors should be able to view and update patients’ medical records. | 1 |  | Bob Dylan, Mick Jagger |
| R\_D\_2 | A way of managing appointments using a dynamic timetable | Doctors provide their availability information and view their daily agenda. | 2 |  | Bob Dylan, Mick Jagger |
| R\_D\_3 | A way of monitoring staff performance. | Doctors or other staff members can compile reports on a weekly or treatment basis. | 2 |  |  |
| 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 | Scheduling new appointments | The patients should be able to create an issue for a new appointment, providing information about their problem. | 1 |  |  |
| R\_C\_3 | 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\_4 | 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\_5 | 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

*Everything related to the domain that might be needed in the project shall be mentioned in here. Sometimes the domain Requirements might be thought as part of either functional or non-functional requirements.*

# User Scenarios/Use Cases

Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

* Describes a significant business need
* Identifies, documents, and ranks the problem that is driving the scenario
* Describes the business and technical environment that will resolve the problem
* States the desired objectives
* Shows the “Actors” and where they fit in the business model
* Is specific, and measurable, and uses clear metrics for success

Client registration: Scenario 1

1. Finds out about the app in store, installs it.
2. Sees that there doesn’t exist a registration tab
3. Contacts the clinic (contact information provided in app)
4. Goes to clinic
5. Receives his/her first treatment
6. Admin creates account, provides client with log-in information
7. The client logs-in
8. Open his/her profile page

Client registration: Scenario 2

1. Client comes to the clinic
2. Receives his/her first treatment
3. Client is presented with the app
4. Admin creates account
5. Provides client with log-in information
6. The client logs-in
7. Open his/her profile page

Client request: Scenario 3

1. Client logs-in
2. Clicks on home tab
3. Clicks on request an appointment
4. Selects day, treatment type and sends
5. Admin receives request
6. Manages request //another scenario
7. Client receives notification

Client: Scenario 4

1. Client logs-in
2. Clicks on home tab
3. Checks his/her medical file

Client edit: Scenario

1. Client logs-in
2. Goes to their profile page
3. Clicks edit
4. Clicks field to edit
5. Fills form
6. Submits

Doctor registration: Scenario 5

1. Doctor starts job at clinic
2. Provides admin with personal information
3. Admin creates account
4. Provides doctor with log-in information
5. The doctor logs-in
6. Checks his/her profile page

Doctor: Scenario //login and view their agenda

Doctor: Scenario //finishes treatment and create report, add files??, add notes??

Doctor edit: Scenario

1. Doctor logs-in
2. Goes to their profile page
3. Clicks edit
4. Clicks field to edit
5. Fills form
6. Submits

Admin: Scenario //user creation

1. Goes to user management tab
2. Clicks add user
3. Fills form with obtained information
4. Submits

Admin: Scenario

1. Admin receives appointment request from client
2. Checks doctors which handle treatment type requested
3. Checks agenda of these doctors for available hours
4. Creates appointment

Admin user edit: Scenario

1. Admin goes under user management tab
2. Selects/searches user
3. Opens his/her profile
4. Selects field to update
5. Fills form
6. Submits

Admin: Scenario //finance management

Admin: Scenario //logistic management

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case No. | 1 | 2 | 3 |
| Use Case name | Log-In | Request an appointment | Register a new user |
| Overview | Users have the rights to access the system and its functionality. | Clients are able to request an appointment | Admin can create accounts for doctors and users |
| Actors | Users (doctors,clients) | Client, Admin | Admin |
| Pre-conditions | User must have a valid id number | clients should be registered and logged in | Admin need to have some information about client/doctor |
| Scenario flow | 1. Log-in form 2. User enters his id 3. System checks if id is valid 4. System displays users profile | 1. **Client logs-in** 2. **Goes under Home view** 3. **Clicks request an appointment** 4. **decides day** 5. **system receives request** 6. **Admin manages it** | 1. **Admin goes under user management tab** 2. **Clicks add user** 3. **Fills form with information** 4. **Submits** |
| Errors | -Invalid data  -> Denied Access | -no available hours  ->denied appointment  ->postponed for another day?? | **-already registered**  **->Failed registration** |
| Post conditions | After Log-in has been made the user has access to the system. | After appointment is accepted and managed the client is notified | After registering a new user he/she is able to access their accounts |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case No. | 4 | 5 | 6 |
| Use Case name | Manage users | Edit |  |
| Overview | Admin is able to CRUD users profile | Users are able to edit their profile |  |
| Actors | Admin | Users (doctors,clients) |  |
| Pre-conditions | - |  |  |
| Scenario flow | 1.Admin goes under user management tab  2.selects/search any user  3.modifies specific fields | 1. **User logs-in** 2. **Goes to their profile page** 3. **Clicks on edit on specific field** 4. **Fills form for editing** |  |
| Errors |  |  |  |
| Post conditions | Users should see the modifications immediately | Users should see the modifications immediately |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case No. | 7 | 8 | 9 |
| Use Case name |  |  |  |
| Overview |  |  |  |
| Actors |  |  |  |
| Pre-conditions |  |  |  |
| Scenario flow |  |  |  |
| Errors |  |  |  |
| Post conditions |  |  |  |

In here you may define the written user scenarios tested in the UCED Application given to you.

APPENDIX

The appendixes are not always considered part of the actual Requirements Specification and are not always necessary. They may include

* Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
* Supporting or background information that can help the readers of the Requirements Specification;
* A description of the problems to be solved by the system;
* Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the Requirements Specification should explicitly state whether or not the appendixes are to be considered part of the requirements.

1. **Definitions, Acronyms, and Abbreviations**

Define all terms, acronyms, and abbreviations used in this document.

1. **References**

List all the documents and other materials referenced in this document.

1. **Requirements Traceability Matrix**

The following trace matrix examples show one possible use of naming standards for deliverables (FunctionalArea-DocType-NN). The number has no other meaning than to keep the documents unique. For example, the Bargaining Unit Assignment Process Flow would be BUA-PF-01.

For example (1):

|  |  |  |  |
| --- | --- | --- | --- |
| **Business Requirement** | **Area** | **Deliverables** | **Status** |
| BR\_LR\_01  The system should validate the relationship between Bargaining Unit/Location and Job Class.---Comments: Business Process = "Assigning a Bargaining Unit to an Appointment" (Priority 1) | BUA | BUA-CD-01  Assign BU Conceptual Design | Accepted |
| BUA-PF-01  Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BUA-PF-01  Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BR\_LR\_09  The system should provide the capability for the Labor Relations Office to maintain the job class/union relationship.---Comments: Business Process = "Maintenance" (Priority 1) | BUA | BUA-CD-01  Assign BU Conceptual Design | Accepted |
| BUA-PF-02  BU Assignment Rules Maint Process Flow Diagram | ReadyForReview |

For example (2):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BizReqID** | **Pri** | **Major Area** | **DevTstItems DelivID** | **Deliv Name** | **Status** |
| BR\_LR\_01 | 1 | BUA | BUA-CD-01 | Assign BU Conceptual Design | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-DS-02 | Bargaining Unit Assignment DB Modification Description | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-PF-01 | Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-UCD-01 | BU Assign LR UseCase Diagram | ReadyForReview |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-001 | BU Assignment by PC UseCase - Add Appointment and Derive UBU | Reviewed |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-002 | BU Assignment by PC UseCase - Add Appointment (UBU Not Found) | Reviewed |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-006 | BU Assignment by PC UseCase - Modify Appointment (Removed UBU) | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-CD-01 | Assign BU Conceptual Design | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-DS-02 | Bargaining Unit Assignment DB Modification Description | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-PF-02 | BU Assignment Rules Maint Process Flow Diagram | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-UCD-03 | BU Assign Rules Maint UseCase Diagram | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-045 | BU Assignment Rules Maint: Successfully Add New Assignment Rule | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-051 | BU Assignment Rules MaintUseCase: Modify Rule | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-053 | BU Assignment Rules MaintUseCase - Review Assignment Rules | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-057 | BU Assignment Rules MaintUseCase: Inactivate Last Rule for a BU | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UI-02 | BU AssignRules Maint UI Mockups | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-021 | BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Success | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-027 | BU Assignment Rules Maint TestCase: Modify Rule - Success | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-035 | BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Error Condition | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-049 | BU Assignment Rules Maint TestCase: Modify Rule - Error Condition | ReadyForReview |

For example (3):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **BizReqID** | **CD01** | **CD02** | **CD03** | **CD04** | **UI01** | **UI02** | **UCT01** | **UCT02** | **UCT03** | **TC01** | **TC02** | **TC03** | **TC04** |
| BR\_LR\_01 |  |  | X |  | X |  | X |  |  | X |  | X |  |
| BR\_LR\_09 | X |  |  | X |  | X |  |  | X |  | X |  | X |
| BR\_LR\_10 | X |  |  | X |  |  |  |  | X |  | X |  |  |
| BR\_LR\_11 |  | X |  |  |  |  |  |  |  |  |  |  |  |

1. **Organizing the Requirements**

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

**By System Mode**

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

**By User Class**

Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

**By Objects**

Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

**By Feature**

A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

**By Stimulus**

Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

**By Response**

Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

**By Functional Hierarchy**

When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

**Additional Comments**

Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.

Executive Summary

Project overview

Considering the heavy burden of multiple tasks that 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 we intend to create aims at creating an easy-to-use tool for practitioners at every moment of their work-life.

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 visit plan, payments, health history, visits and treatments history and the former with a comprehensive and well-organized set of tools to manage their agenda and logistics.

In order to substantiate our goals, we find it very important to make a strong connection between our user’s needs and our design and 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 the providing of 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 hopefully make the product quite interactive and 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 and 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 for safety purposes.

[Purpose and Scope of this Specification](#_3znysh7)

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 broader 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

Clients should be able to view their medical file, doctors reports, appointments, images.

.