

Sommario

[INTRODUCTION 3](#_Toc13407186)

[PURPOSE 3](#_Toc13407187)

[SCOPE 3](#_Toc13407188)

[GLOSSARY 3](#_Toc13407189)

[Definition 3](#_Toc13407190)

[Acronyms 3](#_Toc13407191)

[Abbreviation 3](#_Toc13407192)

[REFERENCE DOCUMENT 3](#_Toc13407193)

[DOCUMENT STRUCTURE 3](#_Toc13407194)

[ARCHITECTURAL DESIGN 3](#_Toc13407195)

[OVERVIEW 3](#_Toc13407196)

[Server 4](#_Toc13407197)

[Client 4](#_Toc13407198)

[COMPONENT VIEWS 4](#_Toc13407199)

[ALGORITHMS DESIGN 4](#_Toc13407200)

[USER INTERFACE DESIGN 4](#_Toc13407201)

[REQUIREMENTS TRACEABILITY 5](#_Toc13407202)

[IMPLEMENTATION, INTEGRATION AND TEST PLAN 5](#_Toc13407203)

# INTRODUCTION

## PURPOSE

## SCOPE

## GLOSSARY

### Definition

### Acronyms

### Abbreviation

## REFERENCE DOCUMENT

## DOCUMENT STRUCTURE

# ARCHITECTURAL DESIGN

## OVERVIEW

The goal of this chapter is to analyse and describe the architecture implemented in the **QDocs** application.

As you can see in the following figure the architecture implemented was a simple Client-Server one:

[[FIGURE]]

### Server

The server represents the main backend logic, it allows multiple users to interact with it simultaneously and keeping them separated. From the technical point of view, it was developed using the Firebase development platform including the following services:

* **Authentication**: the server can directly handle the authentication of the users and forward the authentication mechanism to other external services like Facebook and Google. It stores all account information such as usernames and passwords.
* **Storage**: It allows users to store their files providing a Cloud Storage service.
* **Database**: It provide also not relational Realtime Database used to keep correspondence among stored files and their users.

It is also used to make the screen storage updates faster since the client application instead of querying the storage, that would require much time, it queries the real-time database that is faster.

* **Cloud** **Functions**: backend logic used to react on files upload/delete operation and keeping the Realtime DB updated.
* **Logging**: It logs all the interaction between users and the server.

#### Database

[[ Database Diagram ]]

### Client

The client is represented by the Mobile Application itself (**QDocs**), it is a Native mobile app implemented for the Android platform. From the architectural point of view, it uses the MVC pattern (Model-View-Controller)

[[ MVC diagram paradigm ]]

[[ MVC applied to QDocs ]]

## COMPONENT VIEWS

[[ Class diagrams in details (methods) and their interactions ]]

## RUNTIME VIEWS

[[ Sequence diagrams (more important) showing interaction among activities (and server)]]

## SELECTED ARCHITECTURAL STYLE AND PATTERNS

[[ describe the main advantages of client-server arch. ]]

[[ describe patterns used such as MVC, firebase event listener ]]

[[ callback android paradigm ]]

[[ abstract pattern (storage adapter) ]]

# ALGORITHMS DESIGN

[[ provide examples of algorithm implemented ]]

# USER INTERFACE DESIGN

[[ provide whole mobile app lifecycle ]]

# REQUIREMENTS TRACEABILITY

[[ use case diagram ]]

[[ describe which activity is associated to each requirement ]]

# IMPLEMENTATION, INTEGRATION AND TEST PLAN

[[ describe how tests are preformed ]]