**Software di Gestione**

**Laboratorio Chimico**

Febbraio 2019

Preparato da:

Danilo Manicuti

Sommario

[1 Introduzione 3](#_Toc1446784)

[1.1 Panoramica 3](#_Toc1446785)

[1.2 Scopi e Obiettivi 3](#_Toc1446786)

[1.3 Scopi 3](#_Toc1446787)

[1.4 Definizioni 3](#_Toc1446788)

[2 Casi d’uso 4](#_Toc1446789)

[2.1 Use Case: 1 4](#_Toc1446790)

[2.2 Use Case: 2 5](#_Toc1446791)

[3 Tecnologie Utilizzate 5](#_Toc1446792)

[3.1 Applicazione Web 5](#_Toc1446793)

[3.1.1 Struttura del progetto 5](#_Toc1446794)

[3.1.2 Back-end 6](#_Toc1446795)

[3.1.3 Front-end 7](#_Toc1446796)

[3.2 Applicazione Mobile 7](#_Toc1446797)

[3.2.1 Struttura del progetto 7](#_Toc1446798)

[3.2.2 Back-end 7](#_Toc1446799)

[3.2.3 Front-end 8](#_Toc1446800)

# Introduzione

## Panoramica

L’applicazione Gestione Laboratorio Chimico è una Webapp che sarà utilizzata dai dipendenti dell’azienda chimica Azienda1 per effettuare richieste di analisi di laboratorio sui campioni necessari alla produzione dei prodotti finiti.

Questo documento fornirà informazioni riguardanti requirements, obiettivi di progetto, scopi, definizioni e casi d’uso.

## Obiettivi

L’obiettivo principale di questo progetto è permettere ai dipendenti di richiedere analisi di laboratorio dalle loro postazioni o mediante dispositivi mobili utilizzando un approccio moderno che superi il semplice utilizzo di fogli elettronici (Excel…) o cartacei permettendo agli stessi di tener traccia dello stato delle loro richieste.

## Scopi

L’applicazione permetterà agli utenti, previa registrazione e successiva autenticazione, di richiedere analisi di laboratorio mediante la compilazione di un Form dalle loro postazioni fisse e di monitorarne lo stato anche mediante l’utilizzo di dispositivi mobili che utilizzano il Sistema operativo Android.

Obiettivo dell’applicazione è anche quello di informare tempestivamente gli attori coinvolti nel processo di analisi sullo stato di quest’ultime mediante l’utilizzo di email automaticamente generate dal sistema sia in fase di richiesta che in fase di completamento dell’attività.

Un ulteriore obiettivo dell’applicazione, è quello di costruire un repository univoco degli esiti delle richieste di analisi in formato pdf in modo da permettere all’utente una veloce e facile ricerca a seconda delle esigenze lavorative.

## Definizioni

* **Applicazione Gestione Laboratorio:** l’applicazione che è descritta in questo documento
* **Richiedente:** Ente che ha richiesto lo sviluppo del programma
* **Utente:** persona che interagisce con l’applicazione (appartenente all’ente Qualità)
* **Caso d’uso**: descrive l’interazione tra il Sistema e un attore. Definisce i diversi scenari che derivano da questa interazione
* **Attore:** persona che interagisce con i casi d’uso
* **Responsabile del laboratorio chimico:** persona che si occupa di effettuare le analisi sui campioni e di pubblicare il report

# Casi d’uso

## Caso d’uso: 1

**Descrizione: Registrazione / Accesso Utente**

Attori: impiegati ente qualità

Registrazione

* L’utente che non dispone di un account di accesso deve cliccare sulla voce “Registrati”
* Il sistema richiede di compilare il Form con i seguenti campi obbligatori
  1. Nome
  2. Cognome
  3. Indirizzo email valido
  4. Password
  5. Conferma Password
* Il sistema mostrerà un Warning e non permetterà la registrazione nei seguenti casi:
  + Indirizzo email non valido
  + Non coincidenza tra Password e Conferma password
* Conclusa la compilazione del Form in maniera corretta, il sistema permetterà la registrazione e quindi il successivo accesso al sistema.

Accesso Utente

* Per poter accedere all’applicazione è necessario avere un account precedentemente registrato.
* Il sistema richiede l’inserimento di:
  + Indirizzo email valido
  + Password
* Dopo aver inserito le credenziali di accesso e cliccato sul pulsante “accedi” si avrà accesso al sistema come utente standard.

## Caso d’uso: 2

**Descrizione: Richiesta Analisi di laboratorio e completamento**

Attori: Impiegati ente qualità e Responsabile Laboratorio Chimico

Richiesta Analisi (Utente)

1. L’utente richiede l’analisi di laboratorio cliccando sul menu “Gestione Richiesta di Analisi”
2. Successivamente dovrà cliccare sulla voce “Crea Richiesta”
3. L’utente dovrà compilare tutti i campi previsti dal Form e scegliere le analisi da effettuare
4. Effettuata la compilazione successivamente, cliccando sul pulsante “Invia Richiesta”, il sistema invierà una notifica via email all’ responsabile del Laboratorio
5. La richiesta effettuata sarà visibile dall’utente cliccando sul menu “Visualizza” e successivamente sulla voce “Richieste Aperte”
6. Al completamento dell’analisi, l’utente potrà consultare il report cliccando sul menu “Visualizza” e successivamente sulla voce “Richieste Chiuse”

Completamento Analisi e caricamento report (Responsabile del Laboratorio Chimico)

1. Il responsabile del laboratorio chimico avrà la possibilità di consultare tutte le richieste a lui pervenute cliccando sul menu “Visualizza” e successivamente sulla voce “Richieste Aperte”
2. Effettuata l’analisi richiesta, il responsabile dovrà caricare il relativo rapporto in formato pdf cliccando sulla voce “Carica Report”
3. Caricato il report, la richiesta verrà considerata chiusa e sarà consultabile da entrambi gli attori cliccando sul menu “Visualizza” e successivamente sulla voce “Richieste Chiuse”
4. Il sistema genererà una email per notificare all’utente che la sua richiesta è stata completata.

# Tecnologie Utilizzate

## Applicazione Web

### Struttura del progetto

La struttura del progetto web è stata costruita separando i files in diverse directory in base alle loro caratteristiche. Di seguito le scelte adottate:

* **Directory root** contiene i seguenti files:
  + **App.js** file di inizializzazione dei moduli di Nodejs
  + **Routes.js** contiene le rotte di Express e l’inizializzazione dei relativi componenti
  + **Dbconn.js** definisce i parametri di connessione al database Mongodb
* **Directory controller** contiene i seguenti files:
  + **Listarichieste.js** definisce i controller e le funzioni associate ai template che permettono la visualizzazione della lista delle richieste aperte e chiuse
  + **Login.js** inizializza Angularjs per il template di login
  + **Mainpage.js** definisce i controller e le funzioni associate alla pagina principale del progetto e le rotte utilizzate dal componente **Ngroute**
  + **Registrazione.js** inizializza Angularjs per il template di registrazione utente
  + **Richiesta.js** definisce i controller e le funzioni associate al template di creazione richiesta
* **Directory css** contiene la lista degli stili associati ai template
* **Directory function** contiene i seguenti file di funzione:
  + **Jwtdecrypt.js** funzione che permette di decriptare un jsonwebtoken per ottenere informazioni riguardanti l’utente: id, gruppo e email
  + **Sendemail.js** funzione che permette l’invio di notifiche email all’inserimento/chiusura di una richiesta di analisi
* **Directory views** contiene i template delle diverse pagine dell’applicazione

### Back-end

* **NodeJs** utilizzato per creare la piattaforma web su cui “gira” l’applicazione. Di seguito i moduli utilizzati per lo sviluppo l’applicazione:
  + **Express** utilizzato per creare il web server necessario per rendere disponibile l’applicazione ai diversi utenti
  + **Ejs** framework per la gestione dei template delle pagine web
  + **Jsonwebtoken** utilizzato come middlewareper proteggere l’accesso alle rotte del web server
  + **Bcrypt** utilizzato per la criptazione delle password di accessoche saranno poi memorizzate sul database
  + **Formidable** utilizzato per effettuare il parsing dei dati provenineti dal form di caricamento report al fine di fare l’upload del pdf sul database
  + **FS** utilizzato per leggere il file temporaneo creato dal modulo **formidable**
  + **Nodemailer** modulo che permette di inviare email utilizzando un server SMTP
* **Mongodb**
  + La scelta di Mongodb come database è stata dettata da due fattori:
    - Integrazione con Javascript (memorizza i dati in documenti flessibili JSON-like)
    - Semplicità di utilizzo e flessibilità

### Front-end

* **Angularjs** la scelta di tale Framework è stata determinata dal fatto che si è deciso di utilizzare un approccio basato sul Single Page Application (SPA) e quindi sul paradigma MVC (Model, View, Controller). Per la gestione delle rotte interne si è utilizzato il componente **ngroute**
* **Bootstrap v4 (CSS)** per quanto riguarda l’aspetto dell’applicazione si è deciso di utilizzare il Framework bootstrap v4 in quanto rappresenta ormai uno standard affermato nella realizzazione di web Application.

## Applicazione Mobile

Per lo sviluppo dell’applicazione mobile si è deciso di optare per un’applicazione di tipo Ibrida mediante l’utilizzo del framework **Ionic v1** e del contenitore web **Cordova**.

Tale versione permette esclusivamente di monitorare lo stato delle richieste aperte e di accedere a quelle chiuse permettendo anche di effettuare il download dei report.

### Struttura del progetto

La struttura del progetto web è stata costruita separando i files in diverse directory in base alle loro caratteristiche. Di seguito le scelte adottate:

* **Directory root** contiene:
  + **Server.js** contienei parametri di inizializzazione dei moduli di nodejs e le rotte per ottenere dati da mongodb
* **Directory css** contiene gli stili associati alle diverse pagine html
* **Directory js** contiene il file **app.js** all’interno del quale sono definiti:
  + Le rotte del modulo ui-router di angularjs
  + I controller e le funzioni associate alle diverse pagine html
  + Servizi custom utilizzati all’interno del progetto
* **Directory www** contiene le diverse pagine html del progetto
* **Directory report** all’interno di tale directory vengono generati i report che saranno poi scaricati dal dispositivo mobile

### Back-end

* **Nodejs** è stato utilizzato per creare un web server che permettesse di ottenere dati da database esterni. Di seguito i moduli utilizzati:
  + **Express** per la creazione dell’interfaccia tra applicazione e database
  + **Bcrypt** per decriptare la password utente memorizzata nel database e verificarla con quella inserita nel form di autenticazione
  + **Fs** per creare all’interno della directory temporanea del webserver una copia del report
* **Ionic** è stato utilizzato per creare il web server principale dell’applicazione
* **Mongodb** nello sviluppo dell’applicazione mobile si è fatto riferimento allo stesso database dell’applicazione web.

### Front-end

* **Ionic v1** gli oggetti di tale framework (ion-content, ion-header…..) sono stati utilizzati per la costruzione delle singole pagine web
* **Angularjs** utilizzato per:
  + Gestire il routing tra le diverse pagine mediante il modulo **ui-router**
  + La gestione dell’autenticazione attraverso:

**Angular-jwt** utilizzato per la verifica del token

**Ngstorage** per memorizzare all’interno del browser il token generato dopo l’autenticazione ($localstorage)

**Servizio $q** utilizzato come middleware all’interno del servizio di gestione delle rotte (mediante la proprietà **resolve** del modulo **ui-router**) per eseguire in maniera asincrona la verifica della validità del token (questo tipo di oggetto è chiamato **Promise**)