
UNIVERSITÀ
DEGLI STUDI
DI PALERMO

# Object Design Document

Progetto a cura di:

*Fabio Villa*
*Riccardo Licari*
*Giovanni Bernardo*
*Emanuele Adolfo Ferrara*

# Indice

# 1.Introduzione

## 1.1 Object design trade-offs

Gli obiettivi che il sistema si propone di avere sono stati trattati all'interno della documentazione RAD
Nella seguente, descriveremo l'approccio pensato per la realizzazione del sistema e i vari riferimenti sugli aspetti dell'Interfaccia, Server e dell'App.
Il pattern architetturale di riferimento è il Three-Tier. Questo pattern è il più adatto se si vuole lavorare con un approccio a moduli. La presenza di layer differenti offre la possibilità di sostituire singole componenti con una maggiore libertà di movimento.

Entrando nel dettaglio della suddivisione in moduli/livelli, vediamo:

- Livello di Interfaccia Grafica (Pagine web)
- Funzionalità di base del sistema (Control)
- Accesso ai dati (DAO)
- DBMS (MySql)

## 1.2 Gestione dell'interfaccia grafica

Per la realizzazione del frontend si è scelto di utilizzare come framework di sviluppo React. React è una libreria Javascript che permette di creare dei componenti interattivi in modo da semplificare la realizzazione del progetto.Ogni componente è in grado di definire il proprio aspetto (ossia il markup), mantenere un proprio stato e racchiudere una parte di logica.
Per lo sviluppo stati utilizzati principalmente i seguenti moduli:

- Reactstrap
- Axios
- availity-reactstrap-validation
- react-counter-input
- history
- localStorage
- react-icons/fa

## 1.3 Gestione componente server

La tecnologia scelta per la realizzazione dei nostri obiettivi lato server è NodeJS, con l'ausilio del framework Express.JS.
Tra i pregi che ci hanno permesso di scegliere NodeJS riconosciamo sicuramente una velocità di implementazione e manutenzione data dall'architettura modulare; inoltre NodeJs supporta lo sviluppo di codice asincrono.

NodeExpress è stato usato per la scrittura delle API del sistema.

### 1.3.1 Connessione al database e pattern DAO

Per interfacciarsi con il database si è utilizzato il modulo MySql di Node.
Questo modulo supporta l'esecuzione simultanea di più query (multi-queering) e, anche per questo, risulta essere molto valido.

Per l'accesso ai dati contenuti all'interno del database si è utilizzato il pattern architetturale DAO. In questo modo si può mantenere un certo livello di stratificazione poichè viene fornita alla sezione dei controller un'interfaccia indipendente dallo specifico RDBMS.
Di seguito elencati alcuni moduli necessari per la realizzazione del progetto:

- **Joi:** Modulo utilizzato per la validazione dei dati
- **fs:** Modulo utilizzato per la gestione dei file all'interno dei server
- **MySql**: Modulo utilizzato l'interfacciamento al DBMS MySql
- **Multier**: Modulo utilizzato per la gestione di file e immagini
- **Moment**: Modulo utilizzato per effettuare operazioni sulle date
- **nodeMailer**: Modulo utilizzato per l'invio automatizzato di email
- **bcrypt:** Modulo utilizzato per la gestione di password crittografate

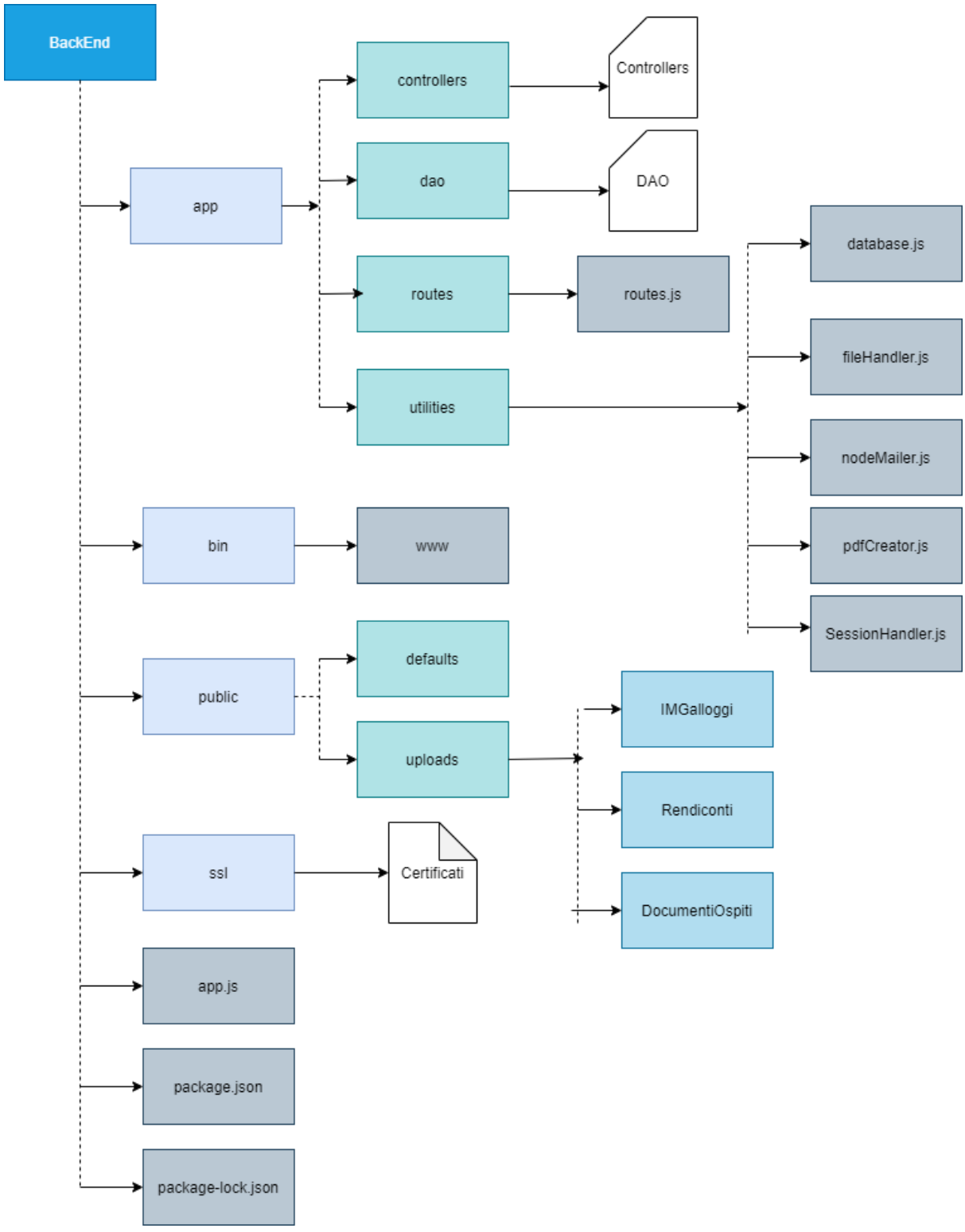
## 1.4 Gestione dell'interfaccia mobile

Per garantire un'accesso facilitato al sistema, da dispositivi mobili, si è pensato di utilizzare una webview: soluzione software associata ad una applicazione android facilmente installabile su un gran numero di dispositivi attualmente in commercio.

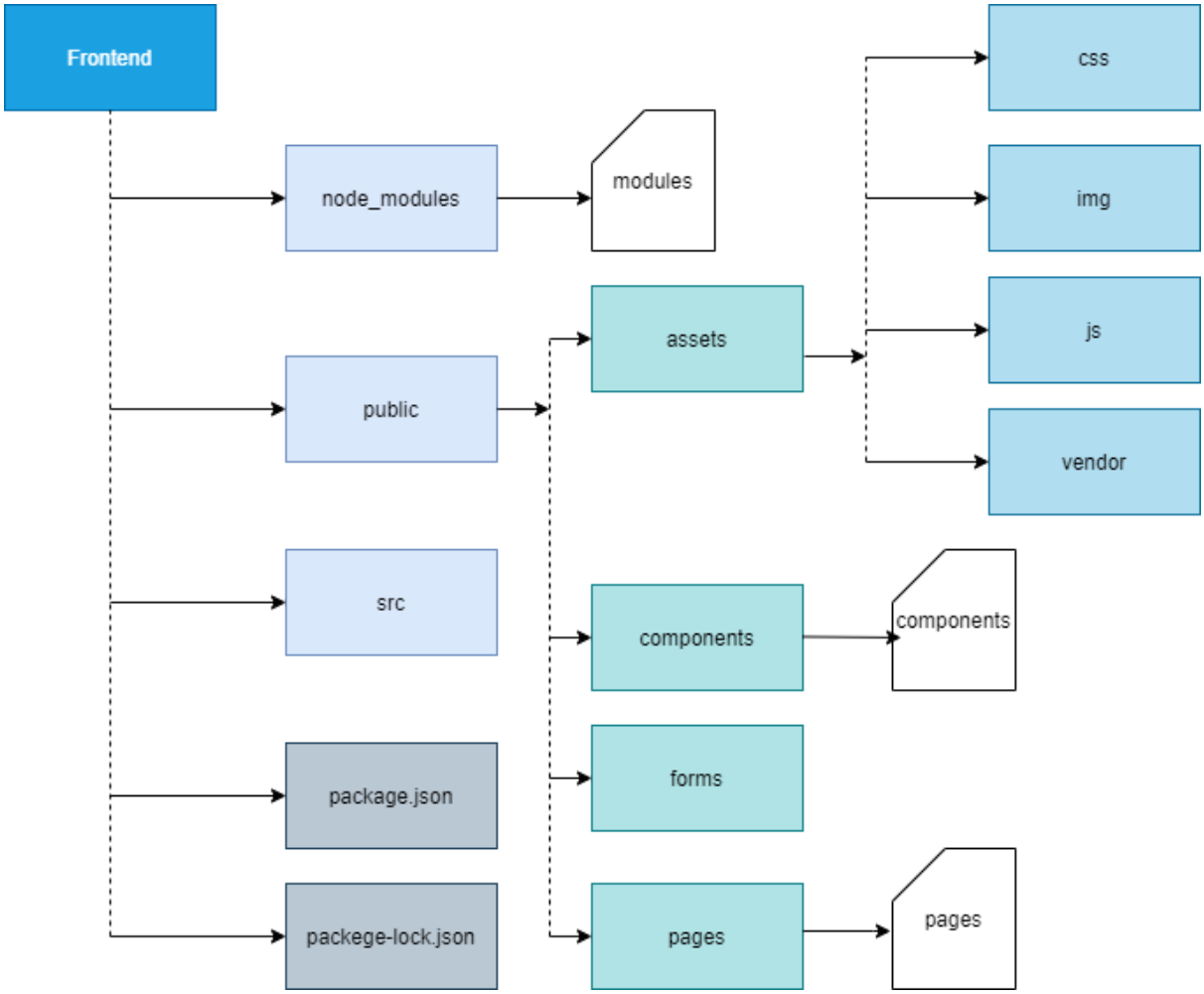
# 2. Packages

Di seguito mostriamo i packages e l'organizzazione dei file sia di backend che frontend in modo da semplificare la lettura e avere chiarezza sull'organizzazione del lavoro svolto

## 2.1 Backend



## 2.2 Frontend


Frontend
node_modules
modules
public
assets
css
img
js
vendor
src
components
components
package.json
forms
packege-lock.json
pages
pages