**“Software Engineering”**

**Course**

**a.a. 2019-2020**

**Lecturer: Prof. Henry Muccini (henry.muccini@univaq.it)**

**#2 - Gestionale per l’occupazione delle aule e delle stanze di studio e lavoro**

|  |  |
| --- | --- |
| **Date** | 05/12/2019 |
| **Deliverable** | Prima Milestone - M1 |
| **Team (Name)** | **VIP** |

|  |  |  |
| --- | --- | --- |
| **Team Members** | | |
| **Name & Surname** | **Matriculation Number** | **E-mail address** |
| **Giacomo Sfratato** | **247318** | **giacomo.sfratato@student.univaq.it** |
| **Bryant Michelle Sarabia Ortega** | **251826** | **bryantmichelle.sarabiaortega@student.univaq.it** |
| **Dawid Pado** | **251813** | **dawid.pado@student.univaq.it** |

# 

# Table of Contents of this deliverable

[**Table of Contents of this deliverable**](#_heading=h.rpw4eatcn7v3) **1**

[**List of Challenging/Risky Requirements or Tasks**](#_heading=h.s7aoej2khc09) **2**

[**A. Stato dell’Arte**](#_heading=h.dzu8s5tg8858) **2**

[**B. Raffinamento dei Requisiti**](#_heading=h.bvvd808qo9na) **3**

[B.1 Servizi (con prioritizzazione)](#_heading=h.a6gjcdoiv0pw) 3

[B.2 Requisiti Non Funzionali](#_heading=h.dkzl9kfpk5w0) 5

[B.3 Scenari d’uso dettagliati](#_heading=h.n1pfjbvz156d) 5

[B.4 Excluded Requirements](#_heading=h.coadjfyv1q4k) 6

[B.5 Assunzioni](#_heading=h.pig7ofu5s251) 6

[B.6 Use Case Diagrams](#_heading=h.uaxum93jk0xq) 7

[**C. Architettura Software**](#_heading=h.wi3ajb5tupem) **8**

[C.1 The static view of the system: Component Diagram](#_heading=h.wectlcat5t4w) 8

[C.2 The dynamic view of the software architecture: Sequence Diagram](#_heading=h.kvvo9r6g924x) 11

[**D. Dati e loro modellazione**](#_heading=h.8cm747oeskxg) **16**

[**E. Design Decisions**](#_heading=h.76541ed23tup) **18**

[Gestione dei Check-in/Check-out](#_heading=h.5q3pf3csv9p3) 18

[Geo-localizzazione delle aule](#_heading=h.57hyrsm2ql70) 18

[Impedire che un utente occupi un posto quando non presente](#_heading=h.poipwzrzkpqr) 18

[Scelta del database e del linguaggio di programmazione](#_heading=h.df69zvtm1g7q) 18

[Apertura/Chiusura Aule](#_heading=h.bjc6x75os8mo) 19

[**F. Design di Basso Livello**](#_heading=h.18j8mykn69gj) **20**

[**G. Explain how the FRs and the NFRs are satisfied by design**](#_heading=h.xroiwcc6o4fj) **21**

[**H. Effort Recording**](#_heading=h.4ph874wqnrpt) **24**

[PERT Chart](#_heading=h.xt0as7l2pyh2) 24

[**Appendix. Prototype**](#_heading=h.ubxlaojvlcp1) **25**

# 

# *List of Challenging/Risky Requirements or Tasks*

|  |  |  |  |
| --- | --- | --- | --- |
| **Challenging Task** | **Date the task is identified** | **Date the challenge is resolved** | **Explanation on how the challenge has been managed** |
| **Gestire il check-in e l’identità degli utenti nelle aule** | **20/11/2019** | **27/11/2019** | Per realizzare il check-in l’utente dovra scannerizzare col proprio dispositivo mobile il codice QR che si troverà sullo schermo di ogni aula, al momento di realizzare il check-in si registrerà anche l’identità dell’utente |
| **Evitare il blocco delle postazioni di lavoro da parte di utenti non presenti** | **20/11/2019** | **27/11/2019** | Per evitare che un utente blocchi una postazione useremo un receiver bluetooth che nel momento di perdere la connessione libera la postazione di studio/lavoro |
| **Supporto vocale** | **20/11/2019** | **21/11/2019** | Per la prima versione dell’applicazione non riteniamo indispensabile il supporto vocale |
| **Creare l’API** | **22/11/2019** | **04/11/2019** | Abbiamo deciso di creare un’API REST utilizzando Laravel il framework di PHP |

# A. Stato dell’Arte

Abbiamo analizzato i seguenti sistemi:

* Sistema Prenotazione Aule Didattiche e Laboratori di UNIVAQ

sito web: <http://e-learning.univaq.it/monitor/coppito22.php>

<http://aule.linfcop.univaq.it/daySmall.php?area=24>

<http://aule.linfcop.univaq.it/daySmall.php?area=26>

<http://aule.linfcop.univaq.it/daySmall.php?area=28>

* Sistema Gestione Aule del Politecnico di Milano

sito web: <https://www7.ceda.polimi.it/spazi/spazi/controller/RicercaRichiesta.do?evn_ricerca_aule_libere=evento&jaf_currentWFID=main>

<https://www7.ceda.polimi.it/spazi/spazi/controller/RicercaAuleLibere.do?evn_giornaliera=evento&jaf_currentWFID=main>

* Sistema Prenotazione Aule dell’Università di Perugia

sito web:www2.ec.unipg.it

# B. Raffinamento dei Requisiti

## B.1 Servizi (con prioritizzazione)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID Servizio** | **Descrizione** | **Soluzioni** | **Complessità** | **Importanza** |
| 1 | Aprire o chiudere una stanza/aula di studio o lavoro; | L’interfaccia deve poter settare un’aula come chiusa o aperta.  Ove possibile si controllano le porte tramite magneti/serrature smart, altrimenti impiegato che chiude l’aula. | Media | Bassa |
| 2 | Conoscere la disponibilità di ciascuna stanza/aula; | Aula chiusa: non disponibile;  Aula aperta & posti finiti: non disponibile;  Aula aperta & posti disponibili: disponibile; | Bassa | Alta |
| 3 | Prenotare (per un tempo limitato) tale aula; | Un professore prenota un'aula intera, ci dev'essere una giustificazione e potenzialmente un'approvazione, quindi  Modello:  \_ Richiesta  \_ Giustificazione della richiesta  \_ Approvazione della richiesta da parte di un amministratore | Media | Media |
| 4 | Fare check-in nella stanza; | Assunzione:  dev’esserci dell’hardware presente (schermo con QR-code).  **Soltanto frequenza obbligatoria.**  Si genera un codice QR temporaneo e dopo il check in tale codice viene “bruciato” insieme al dispositivo dal quale è stato generato. (Un codice per un dispositivo) | Alta | Alta |
| 5 | Registrare l’identità di chi è dentro l’aula; | Prelevare dati da sistemi già esistenti dove disponibili (esse3, ecc), altrimenti data-entry. Associare tali dati a ogni codice QR che ha fatto check-in | Alta | Bassa |
| 6 | Evitare che utenti possano bloccare postazioni di studio/lavoro quando non presenti; | Due possibilità:  1) Image analytics per individuare sagoma umana alla postazione  2) Device che parla col telefono e la presenza è garantita dalla presenza del telefono (es. lettore bluetooth).  Nel momento in cui il sistema capisce che non c'è una persona, parte un timer (es. 10 min) e scaduto il timer l'utente perde il "diritto" alla postazione | Media | Alta |
| 7 | Avere una vista geo-localizzata delle aule disponibili; | Usare mappe interne/piantine dove indicare le aule | Media | Bassa |
| 8 | Opzionale: supporto vocale |  | Alta | Bassa |

*Descrivete in* ***dettaglio*** *i servizi offerti dal vostro Sistema, insieme a quelli che ritenete siano le soluzioni concettuali necessarie. In questa fase, non fate riferimento ad alcuna tecnologia specifica. Se volete, intervistate stakeholder e collezionate dati dal web o da altre sorgenti. Dovete acquisire una conoscenza avanzata dei problemi associate ai vostri servizi. Assegnate un ID a ciascun servizio. Prioritizzate inoltre I servizi in base a due scale: importanza alta, media, bassa. Complessità alta, media, bassa.*

## 

## B.2 Requisiti Non Funzionali

* Il sistema dovrà minimizzare il data-entry, integrandosi con applicativi esistenti **(ADAPTABILITY)**;
* Il sistema deve essere indipendente da una specifica Università **(PORTABILITY)**;
* Dovrà poter gestire fino a 1000 stanze in 100 edifici diversi, con flussi pari anche a centinaia di utenti per stanza **(PERFORMANCE - RELIABILITY)**;
* Non dovrà in alcun modo permettere a personale non abilitato di entrare in possesso di informazioni sull’identità di chi è in una stanza studio **(SECURITY)**;
* Il sistema deve essere facile e intuitivo da utilizzare, e dovrà essere utilizzabile sia su desktop che dispositivi mobile **(USABILITY - PORTABILITY)**;
* Il sistema deve rimanere in funzione almeno per tutto l’orario di apertura dell’Università **(AVAILABILITY).**

## B.3 Scenari d’uso dettagliati

1. Un utente accede alla home page del sistema, e non effettua il login perché vuole solo verificare la disponibilità dell’aula A1.7 dell’edificio Alan Turing. Vede che è presente nella lista delle aule disponibili e che sarà disponibile fino alle 16:30. Non sapendo dove si trova l’aula, clicca la casella dell’aula e visualizza una piantina dell’edificio e del piano in cui si trova tale aula.
2. Per quanto riguarda l’apertura delle aule ci sono due scenari possibili:

2.1) Le porte vengono aperte/chiuse dal sistema diretto dal personale specializzato, tramite serrature smart; in caso di avaria o emergenza tutte le porte si aprono;

2.2) Le porte vengono aperte/chiuse direttamente dal personale specializzato, che successivamente setta la disponibilità di tale aula;

1. Un docente vuole prenotare l’aula C1.10 di Coppito 2 per il giorno 26/11/2019 dalle ore 8:30 alle 11:30; dopo aver effettuato il login, manda la richiesta al sistema con motivazione “Lezione di Algoritmi”; un amministratore, dopo aver effettuato il login, controlla e accetta la richiesta, e in automatico il sistema manda la risposta al docente; quindi il 26/11/2019 dalle 8:30 alle 11:30 l’aula C1.10 non sarà disponibile.
2. Supponiamo che uno studente di medicina deve confermare la propria presenza ad un corso di anatomia in aula D2.25. Lo studente entra in aula, apre l’applicazione o il sito web, fa login, scannerizza il QR-code presente in aula e alla fine della lezione lo scannerizza nuovamente per effettuare il check out, a quel punto la sua presenza a lezione viene registrata. Se non fa il checkout, il sistema dopo un tot di tempo lo fa da solo ma la presenza non viene registrata (ciò viene detto all’utente per incoraggiarlo a fare il check-out).

Il check-in e il check-out vengono sfruttati dal sistema per conoscere l’identità delle persone in aula e per conoscere il numero di posti disponibili.

1. Uno studente vuole occupare un posto a un tavolo di Coppito 2. Per occuparlo si siede, connette il proprio smartphone al receiver bluetooth e lo lascia sul tavolo.

Dopo un po’ si alza, prendendo il proprio smartphone, e il sistema avvia un timer di x minuti, trascorsi i quali il posto viene settato come di nuovo libero. Lo studente in questione però aveva attivato la pausa (15 minuti di assenza autorizzata), per cui ha 15 minuti in più rispetto al normale timer per tornare al tavolo.

## B.4 Excluded Requirements

Per ora abbiamo escluso il supporto vocale poiché non lo abbiamo ritenuto indispensabile per una prima versione del sistema oltre che per la sua difficoltà di implementazione.

## B.5 Assunzioni

* Assumiamo che ogni utente sia in possesso di uno smartphone con cui accedere al sistema e una connessione internet abbastanza veloce.
* Per la funzione di check-in/check-out dobbiamo assumere che sia presente un hardware adatto, ad esempio uno scanner per codici QR.
* Assumiamo che in ogni tavolo c’è un receiver bluetooth per occupare/liberare i posti

## B.6 Use Case Diagrams

# C. Architettura Software

*<IF RELEVANT, Report here both the static and the dynamic view of your system design, in terms of a Component Diagram, and their related Sequence Diagrams >*

## C.1 The static view of the system: Component Diagram

**Serratura Smart:**

Componente che chiuderà le aule fisicamente.

**Interfaccia Grafica:**

Componente che permetterà all’utente di visualizzare le diverse funzionalità che può utilizzare, dipendendo dall’utente questo potrà visualizzare più o meno opzioni.

**Session Storage:**

Componente che contiene le informazioni sull’utente e le altre informazioni utili.

**Decodificatore:**

E’ la componente che si occupa di decodificare i dati generati dal generatore di informazione.

**Scanner:**

Componente che si occupa appunto di scannerizzare il codice QR che si trova sullo schermo.

**QR Code Generator:**

Questa componente sarà la parte del sistema che genererà i diversi codice QR dinamicamente

**Codifica Informazione:**

Componente che fa il contrario del decodificatore ovvero si occupa di codificare i dati generati dal generatore di informazione.

**Generatore Informazione:**

Questa componente genera l’informazione che sarà dopo codificata e trasformata in codice QR, per esempio il numero dell’aula, il corso e la data.

**Login:**

Questa componente si occupa di inviare i dati alla componente **“autenticazione”** per essere processati.

**Autenticazione:**

Processa i dati ricevuti dall’interfaccia “login” per dopo inviarli alla componente **“Database”** e effettuare i relativi controlli.

**Check-in/out:**

Si occupa di effettuare il check in o check out dell’utente così come registrare la sua presenza.

**Aule disponibili/piantina:**

Questa componente si occupa di mostrare una piantina delle aule disponibili sulla interfaccia grafica dell’utente.

**Richiesta Prenotazione:**

Si occupa delle richieste di prenotazioni fatti dai docenti e registrarle nel database

**Open/close Door:**

Componente che permetterà all’utente con i relativi privilegi di aprire o chiudere un aula remotamente.

**Occupa/libera:**

Questa componente ha come scopo quello di occupare o liberare un posto dopo aver ricevuto un segnale bluetooth.

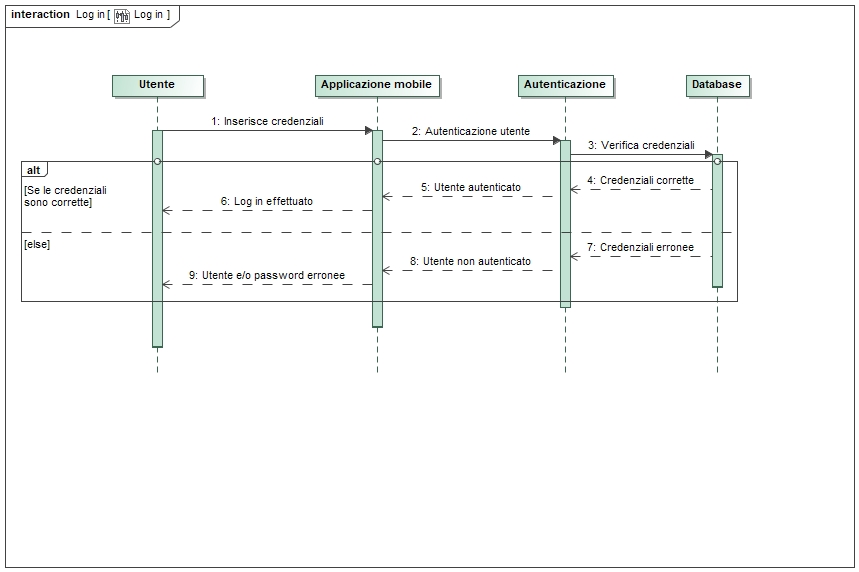
**Posto:**

Comunica al database quando un posto è stato occupato/liberato.

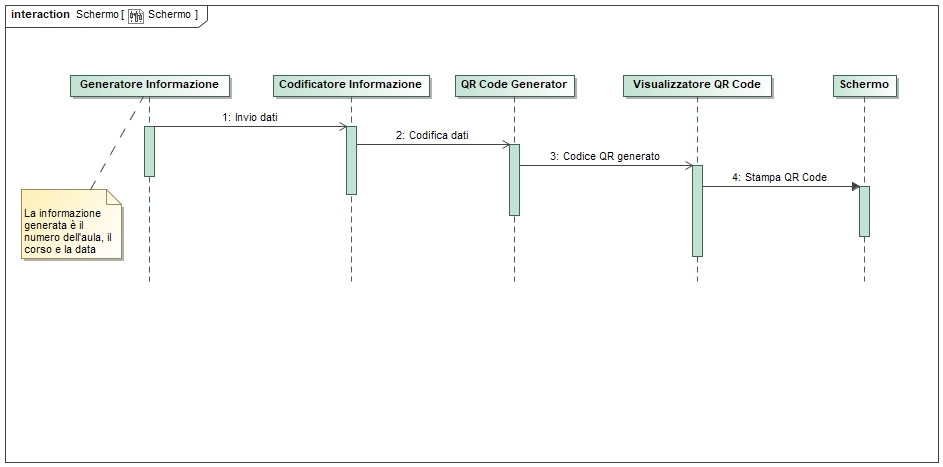
**Database:**

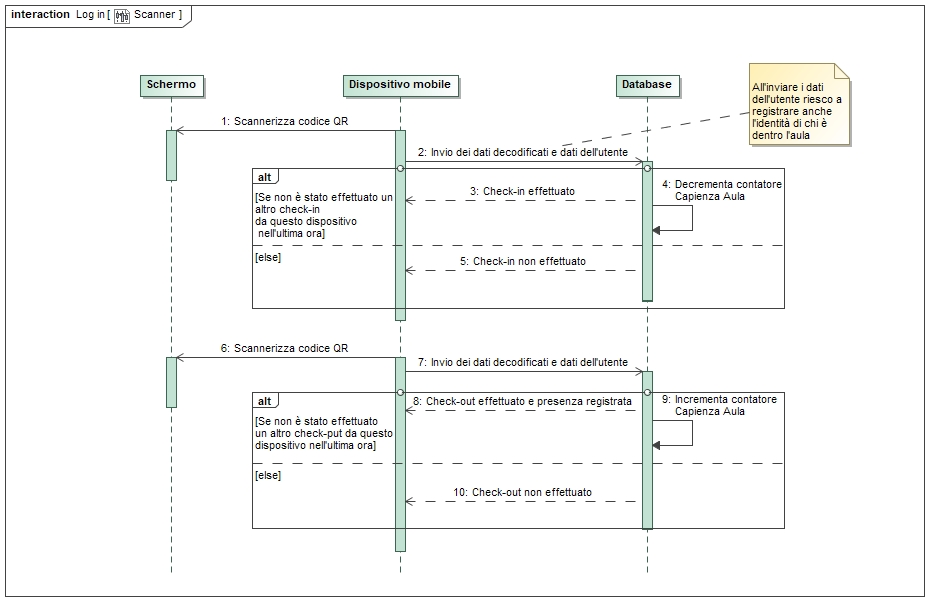
Componente che si occupa di salvare tutti i dati che il sistema gestisce, questo ci permette di sapere in tempo reale la quantità di posti disponibili, le aule disponibili così come l’identità delle persone che si trovano dentro una certa aula, ecc.

## C.2 The dynamic view of the software architecture: Sequence Diagram

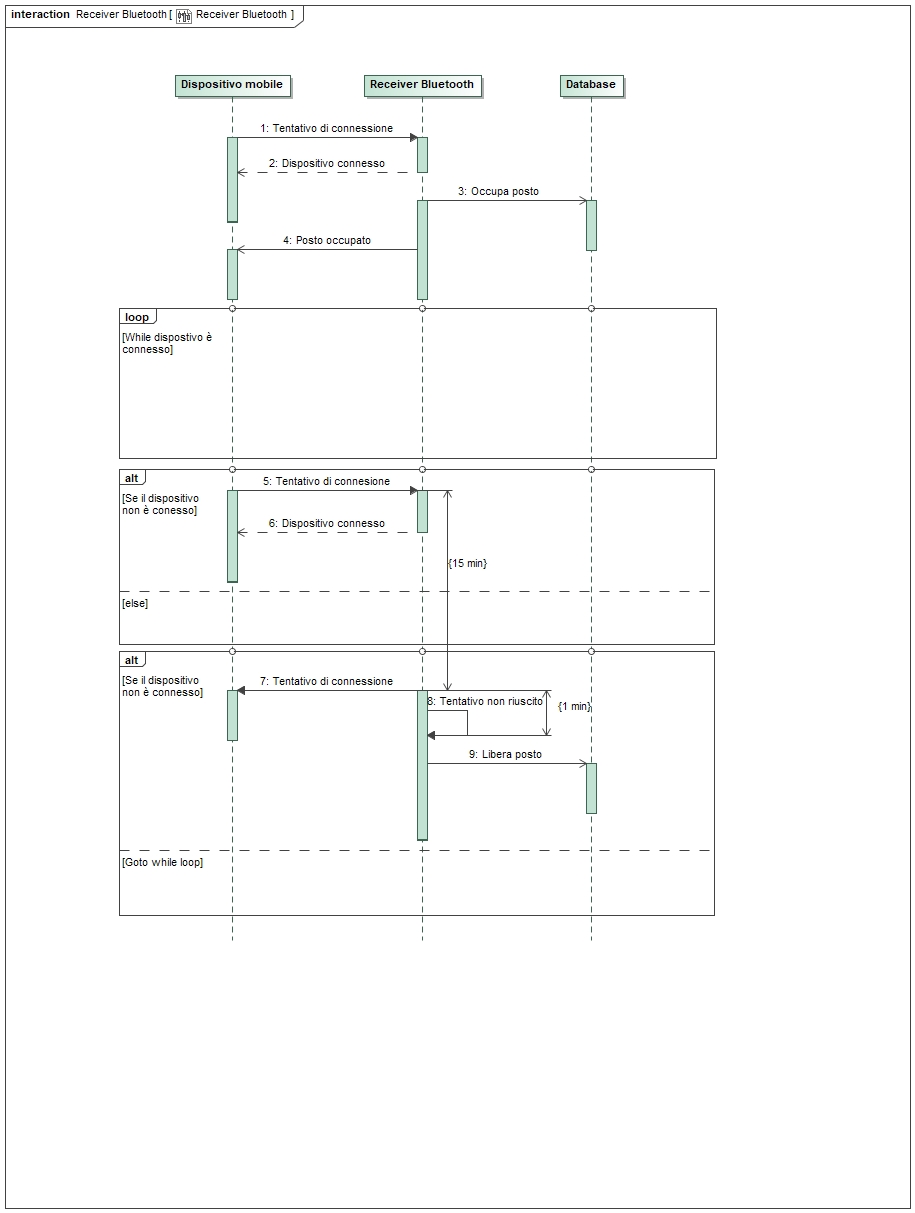


L’utente accede al sistema tramite l’applicazione web, una volta che inserisce il suo account viene effettuata una richiesta di autenticazione al server, questo si connette al database e se risulta presente l’utente visualizza un messaggio di successo sull’applicazione, invece se non risulta presente visualizza un messaggio di errore.

Una volta generati i dati, il codificatore di informazione codifica i dati appena ricevuti. Dopo essere stati codificati si genera un codice QR che verrà visualizzato sullo schermo che si trova nell’aula.

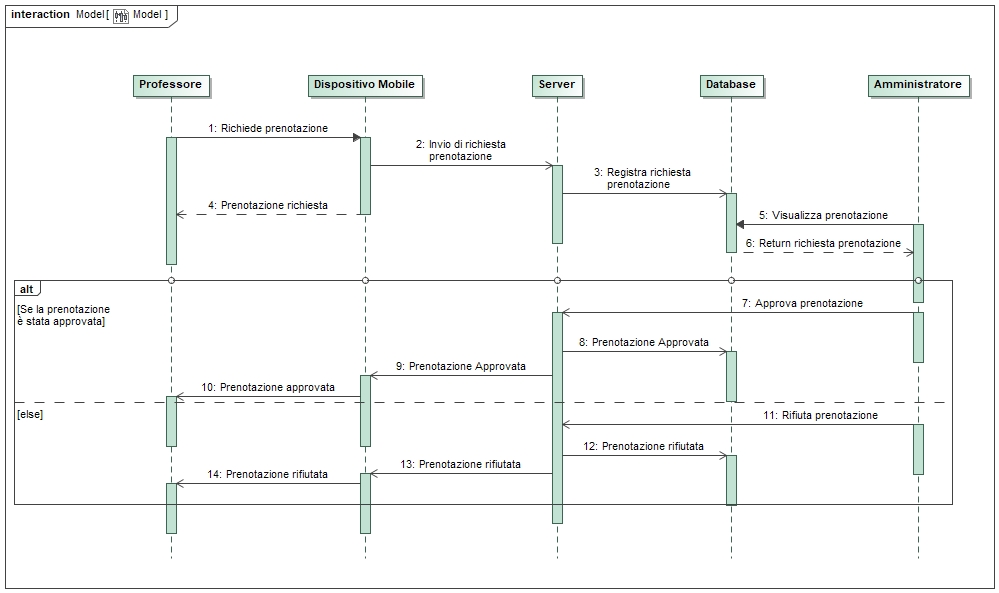


L’utente scannerizza il codice QR che c’è sullo schermo, dopodiché i dati contenuti nel codice QR vengono decodificati e inviati assieme ai dati dell’utente al server, inoltre se non è stato effettuato un check-in da un certo dispositivo mobile nell’ultima ora viene effettuato il check-in e decrementato il contatore “capienza”, altrimenti il check-in non viene effettuato. Stessa cosa per il check-out.



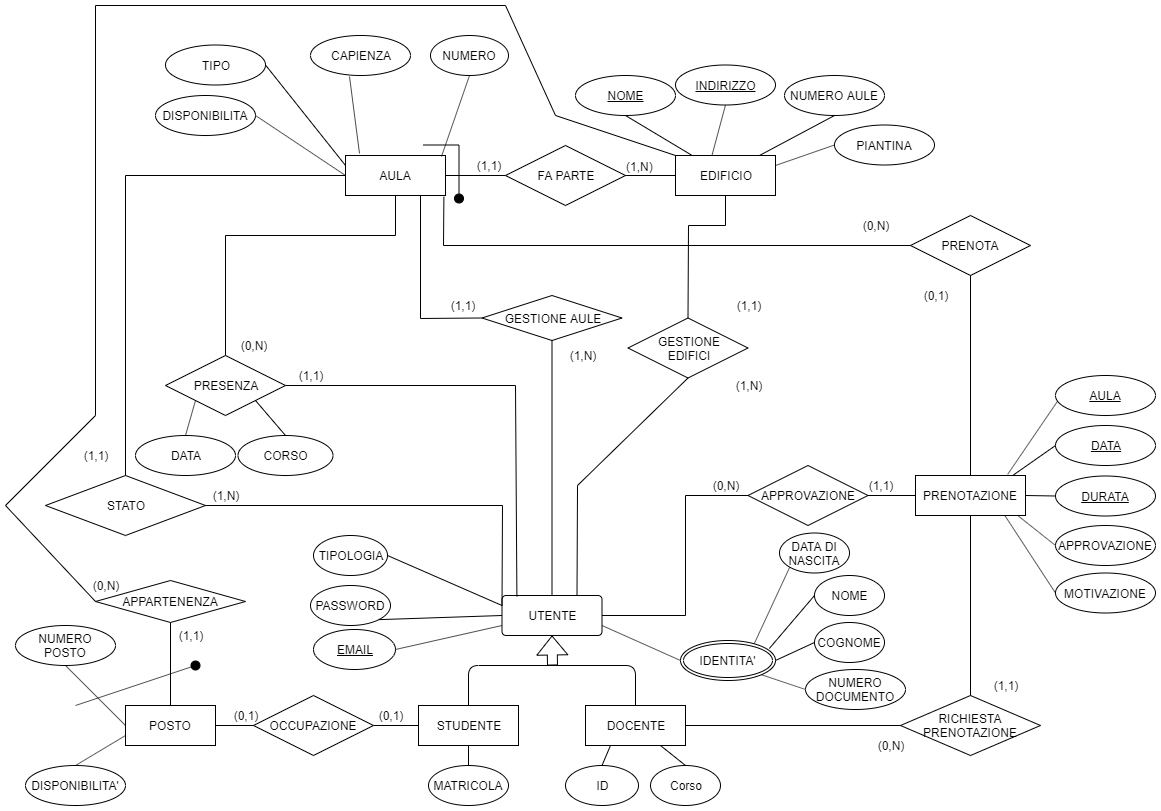
L’utente prima di tutto per occupare un posto si connette al receiver bluetooth, poi il receiver bluetooth manda un segnale al server indicandogli che è stato occupato un posto. Infine viene stampato all’utente un messaggio per indicargli che il posto è stato occupato.

Poi mentre il dispositivo sia connesso ci sarà un loop che non ha nessun effetto. Quando la connessione viene persa però si uscirà dal loop e si effettuerà un tentativo di connessione da parte del bluetooth al dispositivo mobile dell’utente, se questo fallisce allora si setta un timer di 15 minuti per effettuare un ultimo tentativo di connessione, se questo fallisce di nuovo il posto viene liberato, altrimenti si torna al loop.



Il professore prenota un aula dall’applicazione mobile dopodiché viene inviata una richiesta di prenotazione al server che a sua volta viene registrata nel database, infine viene stampato un messaggio sul dispositivo mobile del professore che la richiesta di prenotazione è stata effettuata con successo. Gli amministratori del sistema potranno visualizzare le diverse richieste fatte dai professori sia per approvarle che per rifiutarle, una volta approvata o rifiutata una prenotazione da parte di un amministratore il professore riceve un messaggio di conferma.

# D. Dati e loro modellazione

*Definite le sorgenti di dati a voi necessarie per realizzare I servizi di cui sopra. Modellate tali dati tramite un ER o similari. Specificate se e quali di tali dati sono gia’ forniti da applicativi esistenti*

1. L’entità **AULA** contiene l’informazione delle aule, un’aula appartiene solo ad un edificio, può essere inserita/rimossa da solo un utente, possono essere presenti 0 o più studenti, può essere aperta/chiusa da solo un utente e può avere 0 o più prenotazioni.
2. Un **EDIFICIO** può avere 0 o più aule, può essere inserito/rimosso da solo un utente e può avere da 0 a più posti.
3. L’entità **UTENTE** possiede due figli e un attributo **tipologia** che gli si può assegnare o “**Amministratore**” o “**Dipendente**”.

**-UTENTE AMMINISTRATORE**: Può inserire/rimuovere 1 o più aule/edifici e può aprire/chiudere 1 o più aule.

**-UTENTE DIPENDENTE**: Può aprire/chiudere 1 o più aule.

**-STUDENTE**: Può essere presente solo in 1 aula alla volta e può occupare/liberare

0 o 1 posto.

**-DOCENTE**: Può essere presente solo in 1 aula alla volta e può richiedere la prenotazione di 0 o più aule.

1. L’entità **PRENOTAZIONE** possiede i dati relativi a una richiesta di prenotazione e se è stata approvata o rifiutata. Può essere stata richiesta da solo 1 docente e approvata da solo 1 **UTENTE AMMINISTRATORE**.
2. Un **POSTO** può essere occupato/liberato da 0 a 1 studente e appartenere a solo 1 edificio.

I dati come:

* le informazioni sulla identità degli utilizzatori del sistema
* informazioni sugli edifici, aule e le rispettive piantine

verranno fornite dal database di ciascuna università che vorrà integrarsi.

# 

# 

# E. Design Decisions

*<Document here the 5 most important design decisions you had to take. You can use both a textual or a diagrammatic specification.>*

## Gestione dei Check-in/Check-out

Sin da subito abbiamo dovuto decidere come implementare un meccanismo di check-in nelle aule, e anche il relativo check-out.

Abbiamo infine optato per collocare dei piccoli schermi dentro ogni aula, dove viene trasmesso un codice qr (temporaneo, cioè cambia ogni 3 secondi) e gli studenti con i propri smartphone, dall’app o dal sito web dovranno scannerizzare tale codice per effettuare il check-in nell’aula.

Al termine della lezione dovranno scannerizzarlo di nuovo per effettuare il check-out e registrare così la propria presenza a lezione. Se non dovessero farlo, dopo un tot di tempo il sistema effettuerà il check-out automaticamente ma la presenza dello studente a lezione non verrà registrata.

## Geo-localizzazione delle aule

Inizialmente avevamo pensato di utilizzare le api di google maps per mostrare una vista geolocalizzata delle aule, ma poi abbiamo optato per una soluzione più semplice, ovvero utilizzare le planimetrie interne degli edifici evidenziando sopra di esse le aule disponibili o non disponibili ai vari piani.

## Impedire che un utente occupi un posto quando non presente

Abbiamo pensato a varie soluzioni tra cui:

* rilevamento di una sagoma attraverso le telecamere.

(X) Scartata per il costo complessivo elevato e difficoltà di realizzazione

* Localizzazione del cellulare di un utente attraverso GPS

(X) Scartata per elevata imprecisione

* Accertamento presenza del cellulare di un utente attraverso Bluetooth

(✔) Accettata, pochi punti a sfavore

## Scelta del database e del linguaggio di programmazione

I linguaggi di programmazione saranno Javascripts/HTML/CSS per il front-end e PHP per il back-end, in quanto riteniamo un applicativo web più veloce e reattivo di uno realizzato in Java.  
Per quanto riguarda il database abbiamo deciso di utilizzare MySQL poiché di facile utilizzo e implementazione.

## Apertura/Chiusura Aule

Abbiamo stabilito che solo personale autorizzato può aprire e chiudere le porte.

Durante questa operazione l’informazione sullo stato di apertura verrà caricato nel database. In presenza di serrature smart le porte si apriranno/chiuderanno automaticamente, altrimenti lo si dovrà fare fisicamente.

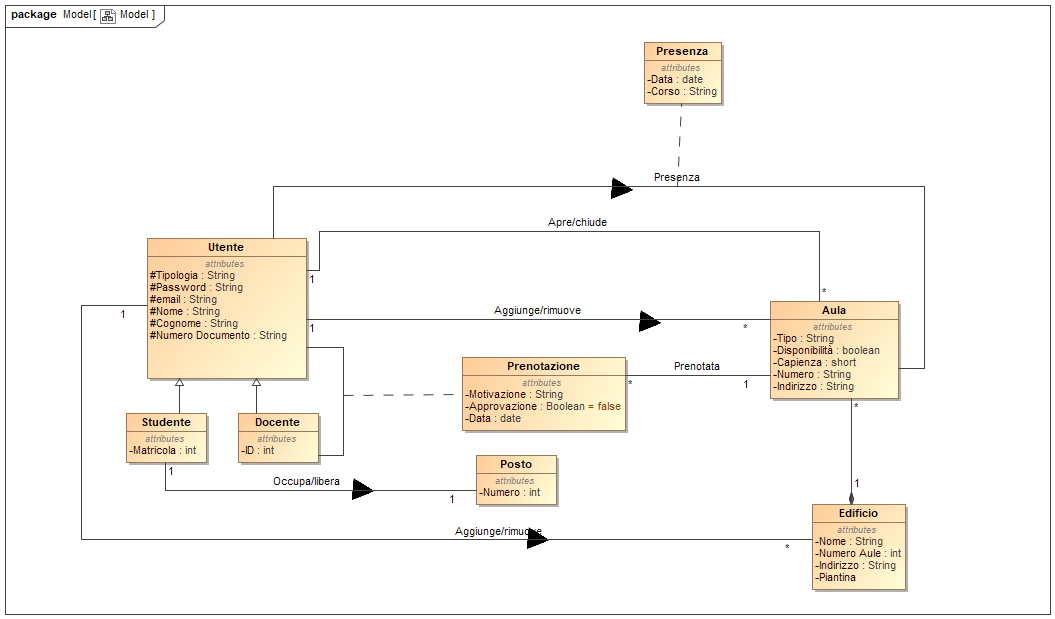
# 

# 

# 

# 

# F. Design di Basso Livello



# G. Explain how the FRs and the NFRs are satisfied by design

*<Report in this section how the architectural and low level design you produced satisfies the FRs and the NFRs>*

* **REQUISITI FUNZIONALI**

1. **Aprire o chiudere una stanza/aula di studio o lavoro**

L’apertura e chiusura delle potrà essere fatta da un utente autorizzato inviando usando l’interfaccia grafica, e premerà un bottone che invierà un segnale sia al server che alla serratura smart di apertura o chiusura

1. **Conoscere la disponibilità di ciascuna aula/stanza**

Usando l’interfaccia grafica la quale prenderà i dati dal server sulle aule disponibili, l’utente potrà visualizzarle in real-time.

1. **Prenotare (per un tempo limitato) un aula**

l’utente potrà prenotare un’aula tramite l’interfaccia grafica la quale invierà la richiesta di prenotazione con il numero di aula, la motivazione, la durata, ecc… al server che dopo sarà approvata o rifiutata da un amministratore garantendo così che un utente non possa prenotare un'aula per un tempo indeterminato.

1. **Fare check-in nella stanza**

Dal nostro design il check-in verrà effettuato tramite lo scanning di un codice QR, il quale insieme a un secondo scanning (che indicherà il check-out) registrerà la presenza dello studente a lezione.

1. **Registrare l’identità di chi è dentro l’aula**

L'identità di una persona che entra dentro un aula verrà registrata grazie all’operazione di check-in.

1. **Evitare che utenti possano bloccare postazioni di studio/lavoro quando non presenti**

La componente bluetooth si incaricherà di liberare le postazioni di lavoro per evitare che siano bloccati quando l’utente non è presente

1. **Avere una vista geo-localizzata delle aule disponibili**

Utilizzando l’interfaccia grafica l’utente avrà la possibilità di visualizzare una piantina dell’edificio in cui saranno visibili tutte le aule disponibili

* **REQUISITI NON FUNZIONALI**

**Performance**

L'obiettivo è quello di gestire fino a 1000 stanze con flussi pari a centinaia di utenti per stanza.Ogni volta che un utente fa sia check-in che check-out i dati di questo utente verranno inviati e registrati nel database perciò il database dovrà essere in grado di rispondere a centinaia di richieste di check-in/check-out alla volta nel minore tempo possibile garantendo così una buona performance e brevi tempi di risposta. Inoltre, da come abbiamo assunto gli utenti devono avere a disposizione una buona connessione internet per facilitare la velocità alla quale i dati vengono inviati al server garantendo una risposta veloce da parte del sistema.

**Portability**

Per quanto riguarda la portability, abbiamo deciso di implementare la prima versione del nostro sistema tramite un applicativo web che potrà essere visualizzato da qualsiasi navigatore che si complementerà con un'applicazione mobile, infine si utillizzerà l'API per inviare, ricevere o aggiornare i diversi dati sul database grazie a JSON.

Infine il sistema funzionerà correttamente nei database supportati dal Query builder di Laravel.

**Adaptability**

Il sistema potrà essere utilizzato da qualsiasi università, riempiendo il database con i dati necessari per il corretto funzionamento del sistema.

**Usability**

Il sistema si potrà adattare facilmente a diversi hardware grazie all'applicativo web che potrà essere utilizzato sia da mobile che da desktop.

**Security**

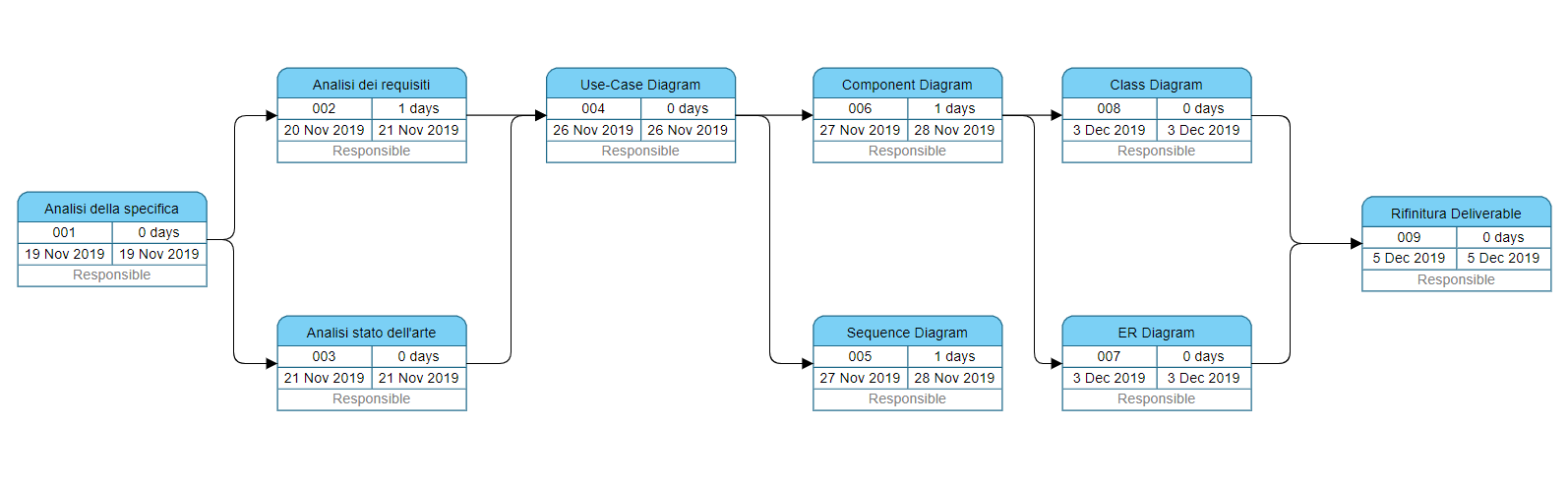
Da come abbiamo disegnato il sistema solo gli utenti che hanno l’attributo tipologia settato a “Amministratore” potranno sapere l'identità delle persone che sono dentro una certa aula, aggiungere/rimuovere aule o edifici evitando così che utenti non autorizzati e senza privilegi possano possano entrare in possesso di dati sensibili e manipolare il database

**Availability**

Il server dovrà stare attivo almeno durante l’orario di apertura e chiusura dell’università per garantire il corretto funzionamento dell’applicazione mobile e le diverse funzionalità, fuori di quest’orario sarà possibile effettuare delle modifiche/aggiornamenti al software.

# H. Effort Recording

## PERT Chart

In linea di massima abbiamo rispettato i tempi che avevamo previsto nel PERT diagram

***Categorization*** *When logging the time spent on the project, please create different sub- categories.**Specifically, it is important to clearly distinguish between two main categories: the time spent for “****learning****” (the modeling languages, the tools, etc.) from the time needed for “****doing****” (creating the models, taking the decisions, …). Learning tasks are in fact costs to be paid only once, while doing costs are those that will be repeated through the project.*

*For each category, please define sub-categories. Examples follow. You may add other sub-categories you find useful.*

|  |  |
| --- | --- |
| ***Learning***   * ***Requirements Engineering*** * ***Non functional Requirements*** * ***Use Case Diagrams*** * ***Tool study*** | ***Doing:***   * ***Requirements discovery*** * ***Requirements Modeling (UC diagrams)*** |

***Summary Statistics****Based on the attributes defined above, calculate the summary statistics of the time spent for “learning”, the time spent for “doing”, and the total time.*

***Note: this Deliverable report shall document only the Summary Statistics for the different deliverables (D1, D2, and Final). Detailed information shall be reported in the Excel file.***

***COPY HERE (computed from the spreadsheet): i) the total number of hours spent by the group (that is, hours per task X number of people working on that task), ii) the time spent for LEARNING and for DOING***

***TOTAL HOURS: 53 hours***

***LEARNING: 1.5 hours***

***DOING: 51.5 hours***

# Appendix. Prototype

*<Provide a brief report on your prototype, and especially: information on what you have implemented, how the implementation covers the FR and NFR, how the prototypes demonstrates your project correctness with respect to the FR and NFR. You may add some screenshots to describe what required above. Be ready to show your prototype during the oral examination>*