Università degli Studi di Salerno

Corso di Ingegneria del Software

System Design Document 2.0

A.A. 2025/26

Progetto: EarthLocals	Versione: 2.0
Documento: System Design Document	Data: 19/01/2026

Coordinatore del progetto:

Nome	Matricola

Partecipanti:

Nome	Matricola
Abbatiello Simone	0512119659
Niemiec Francesco	0512118999
Rega Maristella	0512119032
Squitieri Andrea	0512119008

Scritto da:	Abbatiello Simone, Niemiec Francesco, Rega Maristella, Squitieri Andrea

Revision History

Data	Versione	Descrizione	Autore
16/11/2025	0.1	Stesura iniziale del System Design Document	Abbatiello Simone, Niemiec Francesco, Rega Maristella, Squitieri Andrea
17/11/2025	0.2	Aggiunta gestione dei dati persistenti	Abbatiello Simone, Squitieri Andrea
20/11/2025	0.3	Aggiunta Overview, scopo del sistema ed acronimi	Niemiec Francesco
21/11/2025	0.4	Definizione decomposizione in sottosistemi	Abbatiello Simone, Rega Maristella, Squitieri Andrea
23/11/2025	0.5	Definizione Mapping Hardware/Software	Niemiec Francesco, Rega Maristella
24/11/2025	0.6	Definizione Obiettivi di Design	Niemiec Francesco
24/11/2025	0.7	Modifica alla decomposizione in sottosistemi	Niemiec Francesco, Squitieri Andrea
24/11/2025	0.8	Definizione servizi dei sottosistemi	Squitieri Andrea
25/11/2025	0.9	Revisione Mapping Hardware-Software	Simone Abbatiello
25/11/2025	1.0	Correzione errori di battitura vari e aggiustamenti finali	Abbatiello Simone, Niemiec Francesco, Rega Maristella, Squitieri Andrea
03/01/2026	1.1	Risolti typos	Rega Maristella
07/01/2026	1.2	Sostituzione metodi Access Control List	Niemiec Francesco, Squitieri Andrea
12/01/2026	1.3	Completamento metodi	Niemiec Francesco
19/01/2026	2.0	Correzioni finali	Abbatiello Simone

Contenuti

1 Introduzione

1.1 Scopo del sistema

Il volontariato è una risorsa indispensabile per il benessere collettivo, offrendo numerosi benefici sia a livello individuale che sociale.
Tuttavia, partecipare a esperienze simili al di fuori della propria comunità spesso è ostacolato da barriere logistiche e burocratiche che rendono il processo estremamente complesso e poco accessibile.
Le piattaforme online esistenti per la ricerca di esperienze di volontariato sono in gran parte a pagamento o non offrono una gestione centralizzata che semplifichi la comunicazione tra volontari e organizzatori. In particolare, i costi elevati risultano scoraggianti per chi non dispone di grandi risorse finanziarie.

EarthLocals è un progetto che intende rivoluzionare l'esperienza di volontariato a livello globale,

permettendo a chiunque di contribuire alle comunità mondiali in cambio di ospitalità. La piattaforma vuole fungere da ponte tra organizzatori e volontari, offrendo un'infrastruttura per la comunicazione tra le parti, semplificando così il processo di ricerca di esperienze e l'invio candidature ed eliminando problematiche logistiche che coinvolgono ambo le figure. Di fatto per gli organizzatori la piattaforma faciliterà il processo di raccolta candidature e di ricerca volontari. Invece per quanto riguarda i fruitori del servizio, sarà finalmente dato loro un meccanismo per trovare le esperienze più adatte a loro e candidarsi direttamente a esse.

1.2 Obiettivi di design

Il sistema è progettato facendo riferimento ai seguenti obiettivi:

1.2.1 Obiettivi di performance

- **OP1.0**: Il sistema deve poter essere accessibile da parte di qualunque dispositivo, stazionario o mobile, che permetta l'accesso al Web.
- **OP2.0**: Il sistema deve essere reattivo e avere tempi di risposta brevi per mantenere una certa fruibilità. Sono accettabili dei tempi di attesa leggermente più lunghi solo nel caso in cui il numero di utenti contemporaneamente connessi sia molto alto.

1.2.2 Obiettivi di affidabilità

- **OA1.0**: Il sistema deve proteggere i dati degli utenti durante la fruizione dei servizi. Sarà assicurata una comunicazione sicura tra utente e piattaforma così da evitare fughe di dati.
- **OA2.0**: Il sistema deve avere capacità sufficiente per poter reggere picchi di traffico improvvisi.
- **OA3.0**: Il sistema deve poter gestire correttamente input errati.

1.2.3 Obiettivi di manutenzione

- **OM1.0**: Il sistema deve essere facilmente modificabile, in modo da poter correggere possibili errori in maniera semplice ed essere più facilmente mantenibile.

1.3 Definizioni, acronimi e abbreviazioni

Nel seguente documento sono presenti diversi acronimi, qui riportiamo il significato di ciascuno di essi:

- MVC: Model View Controller.
- DB: Database.
- JDBC: Java DataBase Connectivity.
- RAD: Requirements Analysis Document.
- PS: Problem Statement.
- ORM: Object Relational Mapping

1.4 Riferimenti

Per la stesura di questo documento si fa riferimento ai termini utilizzati ed ampiamente descritti nei documenti di Problem Statement e Requirements Analysis.

Overview

Il seguente documento riporta i dettagli tecnici del design del sistema "EarthLocals".
Dettagli più generali relativi al sistema sono disponibili nel documento di Problem Statement (PS), invece tutto ciò che concerne caratteristiche e funzionalità è disponibile nel documento di Requirements Analysis (RAD).
In questo documento viene proposta una decomposizione del sistema in sottosistemi ed un mapping hardware/software di essi. La gestione dei dati persistenti è presente nel documento "DatiPersistenti_EarthLocals".

Sono inoltre riportati dettagli sul controllo degli accessi e della sicurezza del sistema. Infine, descriviamo il controllo generale del software.

2 Architettura Software Attuale

Allo stato attuale, non esiste alcuna architettura software.

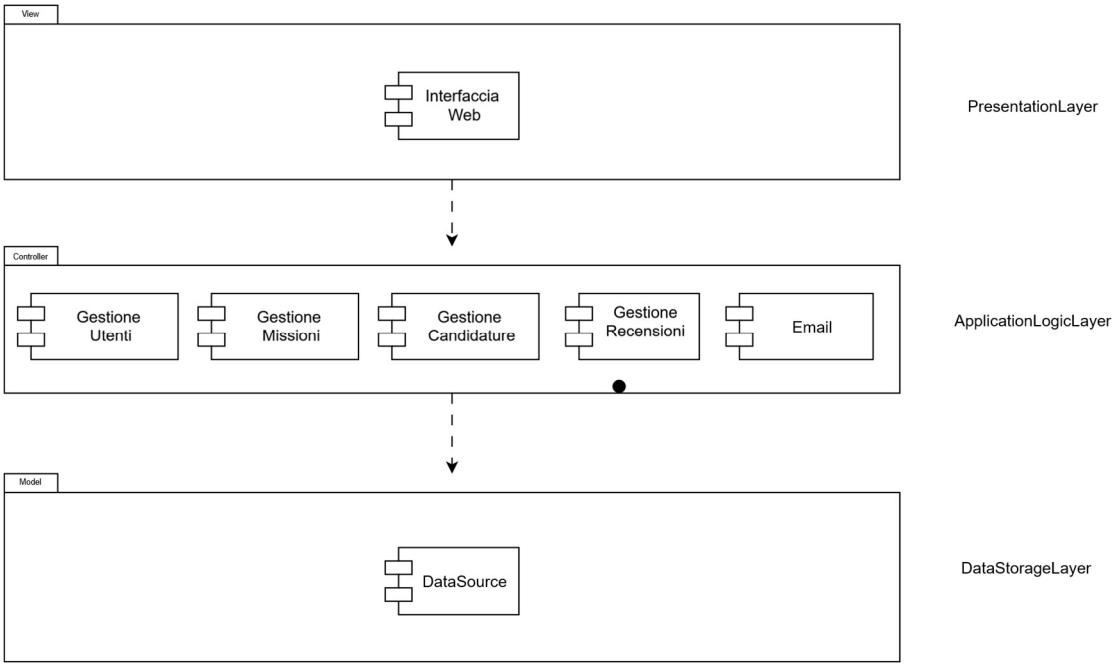
3 Architettura Software Proposta

Overview

Per il nostro sistema abbiamo deciso di utilizzare l'architettura Model View Controller, spesso applicata contestualmente allo sviluppo di applicazione web e GUI.
Questa scelta è fatta per separare la logica dell'applicazione da ciò che l'utente vede e come interagisce con esso.
Esso prevede una divisione in tre sottosistemi principali:

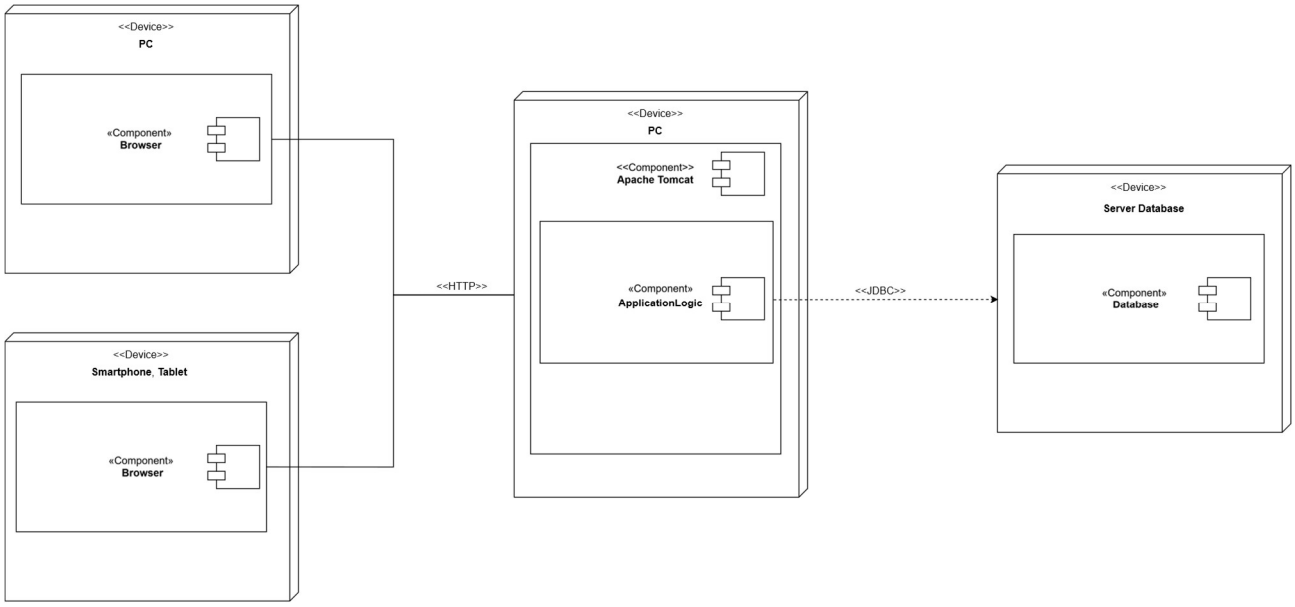
- Model: sottosistema responsabile della conoscenza del dominio applicativo. Si occupa della gestione dei dati, quindi della memorizzazione sul database e dell'interazione col DB stesso.
- View: sottosistema responsabile di mostrare gli oggetti del dominio applicativo agli utenti.
- Controller: sottosistema responsabile della sequenza di interazioni con l'utente e di notificare il View dei cambiamenti nel Model.

3.1 Decomposizione in sottosistemi



3.2 Mapping Hardware/Software

L'architettura è basata su un modello client-server, in particolare su un *fat-client model* dove tutta la presentazione dei dati e parte delle elaborazioni sono demandate al client così da non occupare risorse del server.
Per implementare il sistema utilizzeremo il framework Spring che fornisce una serie di vantaggi come la semplicità di configurazione e soluzioni più moderne per far fronte al testing. Inoltre, Spring fornisce il server web Apache Tomcat embedded pronto all'uso. Il front-end sarà composto da pagine HTML abbinate a Bootstrap per lo stile. Le pagine saranno rese dinamiche grazie al template engine Thymeleaf.



3.3 Gestione dei Dati Persistenti

Fare riferimento al file "DatiPersistenti_EarthLocals". È importante specificare che all'interno del nostro sistema tutto il Database sarà creato tramite JPA, che effettua il mapping oggetti-relazioni automaticamente. Le informazioni saranno estratte grazie ai repository implementati da Spring che forniscono di base i metodi CRUD.

3.4 Access control & Security

	Visitatore	Moderatore	Volontario	Organizzatore	Gestore degli account
Utenti	registerVolunteer, registerOrganizer	Login ,editUser, editPassword, activateAccount	Login, editUser, editPassword, editPassport, activateAccount	Login, editUser, editPassword	Login, createModeratore, editUser, editPassword
Missione	getMissioniAperte	getMissioniAperte, getMissioniPending, acceptMissione, rejectMissione,	getMissioniAperte	getMissioniAperte, getMissioniOrganizzatore , registerMissione	getMissioniAperte
Candidatura	-	-	registerCandidatura , removeCandidatura	acceptCandidatura, rejectCandidatura	-
Recensione	-	removeRecensione	createReview, reportReview	createReview, reportReview	-

3.5 Global Software Control

"EarthLocals" è un'applicazione web: le richieste vengono gestite dal Server che si occupa della logica di business. Le informazioni saranno poi inviate ai controller che le renderanno disponibili alle view(HTML abbianato a Thymeleaf) per fornire una visione finale agli utenti usufruitori. Il Web Server Tomcat si occupa automaticamente di gestire la concorrenza.

4 Servizi dei sottosistemi

Qui sono presenti i servizi che ogni sottosistema fornisce nel sistema software.

4.1 Gestione Utenti

Servizio / Operazione	Descrizione e Contesto
registerVolunteer(datiPersonaliVolontario)	Chiamato durante la registrazione per salvare il nuovo volontario.
registerOrganizer(datiPersonaliOrganizzatori)	Chiamato durante la registrazione per salvare il nuovo organizzatore.
createModerator(datiPersonaliModeratore)	Chiamato per creare un nuovo utente meratore.
activateAccount(datiFinalizzazione)	Chiamato per confermare la registrazione dopo il click sul link email.
***verificaAccesso(email, password)**	Verifica le credenziali al login.
editUser(datiModificati)	Aggiorna i dati del profilo utente.
editPassport(volontario, datiPassaporto)	Aggiorna il passaporto dell'utente.
editPassword(utente, datiNuovaPassword)	Gestisce il cambio password, verificando la vecchia e la validità della nuova.
createPasswordResetToken(email)	Avvia il recupero password verificando l'esistenza dell'email.
resetPassword(email)	Completa il recupero della password.
getProfileInformation(idAccount)	Recupera i dati pubblici del profilo di un altro utente (es. organizzatore).
createVerificationToken(Utente utente)	Crea un token di verifica per completare la registrazione.
getPassportFile(Utente utente)	Permette di scarica il file del passaporto.

*il metodo è implementato automaticamente da Spring.

4.2 Gestione Missioni

Servizio / Operazione	Descrizione e Contesto
registerMissione(datiMissione)	Salva la missione nel catalogo con lo stato iniziale Pending.
acceptMissione(missione)	Il moderatore approva la missione. Cambia stato e invia l'email di notifica.
rejectMissione(missione)	Il moderatore rifiuta la missione.
annullaMissione(missione)	L'organizzatore annulla la missione.
getMissioniAperte(paese)	Restituisce la lista delle missioni aperte relative a quel particolare paese.
getMissioniPending()	Restituisce la liste delle missioni che si trovano nello stato pending.

getMissioneById(idMissione)	Recupera i dettagli di una missione (usato anche prima di una recensione).
getMissioniOrganizzatore(idOrganizzatore)	Recupera le missioni dell'organizzatore per la sua area personale.

4.3 Gestione Candidature

Servizio / Operazione	Descrizione
registerCandidatura(missione, volontario)	Crea una nuova istanza di candidatura associando il volontario alla missione scelta.
removeCandidatura(candidatura)	Elimina (o annulla) una candidatura esistente su richiesta del volontario.
acceptCandidatura(candidatura)	Registra l'approvazione del volontario da parte dell'organizzatore.
rejectCandidatura(candidatura)	Registra il rifiuto della candidatura da parte dell'organizzatore.
getRichiesteCandidatura(organizzatore)	Ritorna tutte le candidature per le missioni di un organizzatore.
getEsperienzeVolontario(volontario)	Ritorna tutte le candidature accettate del volontario.
getCandidatureVolontario(volontario)	Ritorna tutte le candidature del volontario.

4.4 Gestione Recensioni

Servizio / Operazione	Descrizione
aggiungiRecensione(missione)	Salva una recensione scritta da un volontario per un organizzatore.
aggiungiRecensioneVolontario(volontario, missione)	Salva una recensione scritta da un organizzatore per un volontario.
rimuoviRecensione(recensione)	Elimina una recensione segnalata ritenuta inappropriata dal moderatore.
approvaRecensione(recensione)	Conferma la validità di una recensione segnalata, mantenendola pubblicata.

4.5 Gestione Email

Servizio / Operazione (da Diagrammi)	Descrizione e Contesto
inviaConferma(id, token)	Invia link di attivazione account.
inviaEmailRecuperoPassword(email, token)	Invia credenziali per reset password.
inviaEmailCandidatura(id)	Notifica nuova candidatura all'organizzatore.

inviaEmailAnnullamentoCandidatura(id)	Notifica annullamento candidatura (dal volontario).
inviaEmailAccettazioneCandidatura(id)	Notifica accettazione candidatura.
inviaEmailRifiutoCandidatura(id)	Notifica rifiuto candidatura.
invioEmailApprovazioneMissione(id)	Notifica approvazione missione.
invioEmailViolazioneRecensione(id)	Notifica rimozione recensione.
inviaEmailCreazioneModeratore(id)	Invia credenziali al nuovo moderatore creato.