TEAM
COMMITTED

Universitá degli Studi di Padova

Specifica Tecnica V0.0

Informazioni sul documento

Nome documento	Specifica Tecnica
Data documento	GG/MM/AAAA
Redattori	Tutti e Nessuno
Verificatori	xxxxxx
Approvazione	xxxxxx
Uso documento	Interno
Lista distribuzione	• *Team Committed* • *Prof. Tullio Vardanega*

Sommario

xxxxxxxxxxx

Diario delle modifiche

Modifica	Autore	Data	Versione
Completato capitolo 11 (Prototipi di interfaccia utente)	Marco Begolo	2012/01/29	V0.3
Sistemazione del file sorgente per fare in modo che includa tutti i capitoli come file a parte, cosí come la prima pagina. Iniziata la stesura del capitolo 3pag	Giorgio Maggiolo	06/12/2011	V0.2
Inizio creazione del documento. Creato lo schema base e iniziato a scrivere l'introduzione e il capitolo sulle comunicazioni. Impostato inoltre lo schema grafico iniziale del documento stesso.	Giorgio Maggiolo	05/12/2011	V0.1

Indice

1 Introduzione

1.1 Scopo del documento

Il presente documento ha lo scopo di definire e mostrare le scelte progettuali, ad alto livello, che il gruppo ***Team Committed*** ha deciso di rispettare per la realizzazione del prodotto. Al suo interno verranno presentati i vari design pattern utilizzati nella creazione del prodotto, la struttura dei packages e le principali classi che li compongono.

1.2 Scopo del prodotto

Il prodotto denominato **SafetyGame** si propone di fornire uno strumento informatico per la gestione delle pratiche di sicurezza sul lavoro in modo dinamico, evitando corsi di formazione che spesso si dimostrano inutili per la poca attenzione prestata dai partecipanti.

Lo strumento si basa sul concetto di **gamification** che comporta competizione tra i dipendenti all'interno delle aziende creando un sano interesse per un argomento delicato come la sicurezza sul luogo di lavoro.

Il sistema è pensato sia per lavoratori che hanno una postazione fissa dotata di PC, sia per quelli che hanno la necessità di spostarsi e che quindi sono forniti di dispositivi mobili. Ad essi verranno poste periodicamente domande, di varia tipologia, le cui risposte comporteranno l'assegnazione di un punteggio generando una classifica aziendale.

1.3 Glossario

Al fine di evitare ogni ambiguità relativa al linguaggio e ai termini utilizzati nei documenti formali, il glossario viene incluso nel file `Glossario-V2.0.pdf`, dove vengono definiti e descritti i termini marcati da una sottolineatura.

1.4 Riferimenti

1.4.1 Normativi

- **Norme di Progetto, v 2.0** (allegato `Norme_di_Progetto_V2.0.pdf`)
- **Analisi dei Requisiti, v 2.0** (allegato `Analisi_dei_Requisiti_V2.0.pdf`)

1.4.2 Informativi

- **Java 6 SDK** Documentation
  `http://docs.oracle.com/javase/6/docs/`
- W3School **XHTML 1.1** Documentation
  `http://www.w3school.com/html/html_xhtml.asp`
- W3School **CSS 3** Documentation
  `http://www.w3school.com/css/default.asp`

- **MySql** Documentation
  `http://dev.mysql.com/doc/refman/5.5/en/index.html`
- **JSP** Documentation
  `http://java.sun.com/products/jsp/syntax/2.0/synstaxref20.html`
- **Tomcat** Documentation
  `http://tomcat.apache.org/tomcat-6.0-doc/index.html`
- *E. Gamma, R. Helm, R. Johnson, J.Vlissides (1995),* **Design Patterns** **,** *USA-Canada, Addison-Wesley.*

2 Design Pattern

Presenteremo qui di seguito i vari design pattern impiegati nella progettazione dell'architettura del progetto SafetyGame.

2.1 Multitier

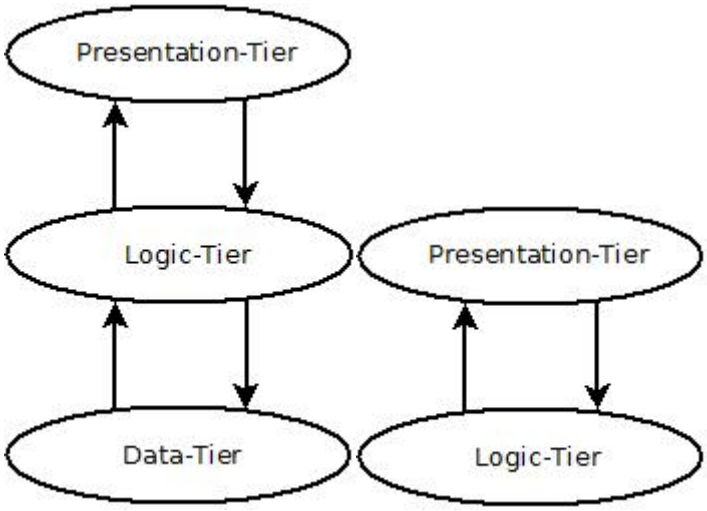

Figura 1: Diagramma dei design pattern ThreeTier & TwoTier

Poichè i design pattern *ThreeTier* e *TwoTier* fanno parte della stessa "categoria" (Multitier), abbiamo deciso di raggrupparli nella stessa sezione. Visto che la descrizione dei due design pattern e le motivazioni dell'adozione di essi sono simili, verranno descritti in questa sezione. Il contesto applicativo, invece, verrà discusso singolarmente, in quanto la loro applicazione è radicalmente differente.

- **Descrizione:** tali design pattern permettono una disgiunzione fra i vari gruppi di entità che cooperano nell'erogazione del servizio. Esisterà un livello che si occuperà di interagire con il cliente offrendo l'interfaccia grafica, un secondo livello per l'esecuzione della parte algoritmica dell'applicazione e, nel caso del *ThreeTier*, un terzo livello per la persistenza dei dati e il loro recupero

- **Motivazione:** il beneficio principale apportato da questo paradigma risiede nel fatto che ogni livello può venir cambiato/aggiornato senza dover propagare gli effetti ai livelli adiacenti.
  Sebbene tale vantaggio derivi principalmente dal raggruppamento delle varie funzionalità in gruppi disgiunti, ma in stretta collaborazione, la struttura di tale design pattern risulta perticolarmente calzante per dei servizi

basati sull'architettura client-server, in quanto ogni livello non esiste semplicemente come raggruppamento logico a sé stante, ma il suo ruolo viene adattato in relazione allo specifico ambiente di rete in cui esegue: nel caso la morfologia della rete cambi, basterà aggiornare lo strato che lavorava nel determinato ambiente

- **Contesto applicativo:**
  - **ThreeTier:** questo design pattern verrà utilizzato come struttura portante del Back-end, in cui:
    1. *Presentation-tier* offrirà l'interfaccia per i vari front-end e si occuperà di reindirizzare le richieste dell'utente alle corrette operazioni
    2. *Logic-tier* si occuperà di eseguire i vari calcoli necessari all'elaborazione della risposta da fornire i dati all'utente
    3. *Data-tier* si occuperà di astrarre la base di dati e della persistenza o il recupero da essa dei dati necessari per eseguire le computazioni richieste
  - **TwoTier:** questo design pattern verrà utilizzato come struttura portante del Front-end Desktop, in cui:
    1. *Presentation-tier* offrirà l'interfaccia grafica agli utenti e si occuperà di reindirizzare le richieste dell'utente alle corrette operazioni
    2. *Logic-tier* si occuperà di gestire la temporizzazione delle domande e la gestione dell'operazione di login

2.2 MVC

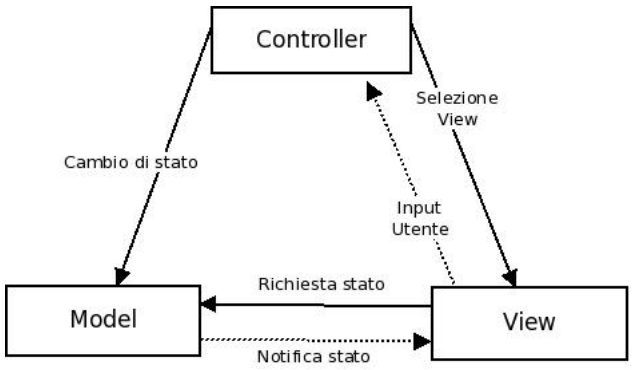

Figura 2: Schema del Design Pattern MVC

- **Descrizione:** *Model-View-Controller* è un pattern architetturale molto diffuso e che permette una completa disgiunzione delle funzionalità che il prodotto deve offrire, organizzando l'applicativo come segue:

  - **Model:** costituisce la parte logica dei dati utilizzati dal sistema, compresa la loro persistenza
  - **View:** costituisce l'interfaccia vera e propria con cui i dati sono visualizzati e tramite la quale gli utenti possono interagire con il sistema
  - **Controller:** rappresenta il componente che riceve i comandi dell'utente e li attua, decidendo le conseguenti operazioni da effettuare nel Model e nel View

  Tale design-pattern è stato scelto in quanto largamente diffuso ed adottato da numerosissime tecnologie moderne. Esso garantisce infatti una solida struttura altamente manutenibile.

- **Motivazione:** l'utilizzo di tale design pattern permette una solida e manutenibile architettura su cui basare l'applicativo

- **Contesto applicativo:** tale design pattern verrà utilizzato per sviluppare il front-end Android ed il front-end web, per dotarli di una struttura ben organizzata e manutenibile. Questa scelta risulta particolarmente importante in un contesto di sviluppo come quello di Android in cui la struttura stessa delle viste (pensate come Activity) può portare ad una mancata separazione dei ruoli.
  A questo scopo sono stati identificati i componenti:

  1. **Model** rappresenta l'interfaccia con cui l'applicativo interagirà con il Back-end tramite richieste HTTP e riceverà informazioni con XML
  2. **View** rappresenta l'effettiva interfaccia grafica tramite la quale gli utenti interagiranno con l'applicativo
  3. **Controller** riceve le scelte dell'utente ed effettua le operazioni corrette sul modello dei dati

  L'architettura MVC dei front-end Android e web è stata pensata per utilizzare un aggiornamento di tipo *pull*. In un applicativo di questo tipo, in primo piano si avrà soltanto una vista alla volta, mentre nel caso in cui un'altra vista che abbia bisogno di accesso ai dati torni in primo piano, questa richiederà un aggiornamento di tali dati.

2.3 Altri Design Pattern

Durante la fase di *Progettazione Architetturale* non sono emersi problemi risolvibili tramite utilizzo di design pattern diversi da quelli già descritti. Sono stati valutati attentamente alcuni design pattern non architetturali, quali *Singleton* o *DAO*, che però non sono stati ritenuti adatti per i problemi che, in fase di *Progettazione Architetturale*, abbiamo affrontato.

Tuttavia, durante la fase di *Progettazione di Dettaglio* potrebbero emergere problemi che saranno risolti facilmente tramite l'uso di design pattern. Per questo motivo ci riserviamo il diritto di introdurre design pattern non architetturali

all'interno della Specifica Tecnica, modificando di conseguenza l'archichettura strutturata attualmente.

3 Strumenti utilizzati

3.1 Java

L'utilizzo di **Java 6** è legato all'approvazione ricevuta dal proponente durante l'incontro avuto con il *Proponente* in data 2011/12/05[1]

- **Vantaggi**
  - **Multipiattaforma:** permettendo l'astrazione dalla macchina fisica tramite la JVM, assicura che l'applicativo risulti funzionante qualsiasi sia il sistema operativo installato sulla macchina che lo esegue[2]
  - **Indipendenza dalle risorse fisiche:** per lo stesso motivo descritto sopra, l'esecuzione dell'applicativo risulterà uniforme indipendentemente dalle risorse fisiche utilizzate
  - **Librerie:** alta disponibilità di librerie presenti in rete che semplificano l'interfacciamento con altri dispositivi e applicativi già esistenti
  - **Compatibilità con piattaforme mobili:** considerata la richiesta di un'applicazione per dispositivi mobili[3], l'utilizzo di tale linguaggio semplifica lo sviluppo dello stesso dell'applicazione mobile
- **Svantaggi**
  - **Lentezza** causata dal passaggio attraverso la JVM e dal fatto di essere un linguaggio interpretato

3.2 XHTML

Si è deciso di utilizzare **XHTML 1.1** assieme a **CSS 3** [4] per la presentazione dell'interfaccia grafica dell'applicativo web sia agli utenti che per gli utenti autenticati.

- **Vantaggi:**
  - **Piattaforma mobile:** poichè la piattaforma deve essere fruibile anche da browser mobile, il semplice XHTML permette di definire agevolmente pagine web che si adattino al browser
  - **Uniformità di stile:** è possibile scrivere facilmente una struttura base dell'applicazione web dove poi andare ad inserire il contenuto tramite JSP
- **Svantaggi:**

[1] Rif: *verbale05122011.pdf*
[2] Requisito *RVOB 1.1*
[3] In particolare Android, come da requisito *RVOB 1.2*
[4] Requisito *RQD 6.2* e *RQD 6.1*

  - **Browser:** è possibile che alcuni browser (sopratutto quelli più datati) non riescano ad interpretare correttamente le informazioni contenute nelle pagine, rendendo difficoltosa, se non impossibile, la lettura delle pagine create con questo linguaggio

3.3 MySql

Per memorizzare tutti i dati che verranno immessi nel sistema SafetyGame, si è scelto di utilizzare la base di dati **MySql Community Server 5.5.20**

- **Vantaggi:**
  - **Facilità d'uso:** poichè durante il corso di Basi di Dati abbiamo studiato i database relazioni (ed in particolare proprio MySql), l'uso di MySql ci è sembrata una scelta quasi obbligata
  - **Relazioni con i vari componenti del sistema:** la documentazione riguardante l'interfacciamento con questa base di dati è di grande quantità, rispetto alle altre basi di dati
- **Svantaggi:**
  - **Prestazioni:** poichè la versione Community Server non è stata pensata per supportare grandi carichi di lavoro[5], quando[6] il database sarà oberato di lavoro e/o conterrà una quantità di dati abnorme le sue prestazioni risulteranno per lo meno fastidiose. Per il momento abbiamo comunque deciso di tenere questa versione, a dispetto della *Enterprise* (ovvero quella a pagamento), perchè come base di dati per un prototipo è più che sufficiente

3.4 Android SDK

L'utilizzo dell'**Android SDK r16** è essenziale per lo sviluppo per l'applicazione mobile su Android

3.5 JSP

JSP 2.1 verrà utilizzato per generare dinamicamente le pagine che compongono l'applicativo web. A JSP si affiancherà l'uso dei *JavaBeans*, che permetteranno la possibilità di estendere agevolmente il codice senza dover creare alcuna nuova entità logica (secondo il modello MVC)

- **Vantaggi:**

[5] Inteso, ad esempio, come migliaia di query nello stesso istante, piuttosto che contenere milioni di record in una singola tabella

[6] N.B. Non se, ma quando

  - **Compatibilità con Java:** poichè gran parte del progetto sarà scritto in Java, l'utilizzo di JSP garantirà, con un certo grado di sicurezza, la compatibilità con i messaggi mandati dagli applicativi Java / Android
  - **Sicurezza:** rispetto ad altri linguaggi di programmazione web (ex. php), JSP ha degli elevati standard di sicurezza interni che permettono di definire le funzionalità del programma senza troppi pensieri riguardanti la sicurezza interna dell'applicazione
  - **Vasta libreria:** poichè basato su Java, JSP integra tutte le librerie native di Java. Questo ci permette, oltre che ad accedere a tutta una serie di codici già scritti, di riutilizzare parte della logica scritta per l'applicativo mobile, ovviamente modificando il codice da Java a JSP

- **Svantaggi:**

  - **Compilazione run-time:** ogni volta che si deve caricare una pagina JSP, se sono state fatte modifiche al contenuto della suddetta, il server dovrà compilare a run-time la pagina, provocando così un piccolo ritardo nel caricamento della stessa

3.6 Tomcat

Tomcat 6.0.35 è un servlet reso necessario dall'uso di JSP.

4 Architettura Generale

L'applicativo è stato progettato come unione di quattro sottoinsiemi, come mostrato in figura 4. Ogni sottosistema è stato inoltre suddiviso in componenti, i quali sono indipendenti tra di loro o, in ogni caso, con un basso indice di accoppiamento.

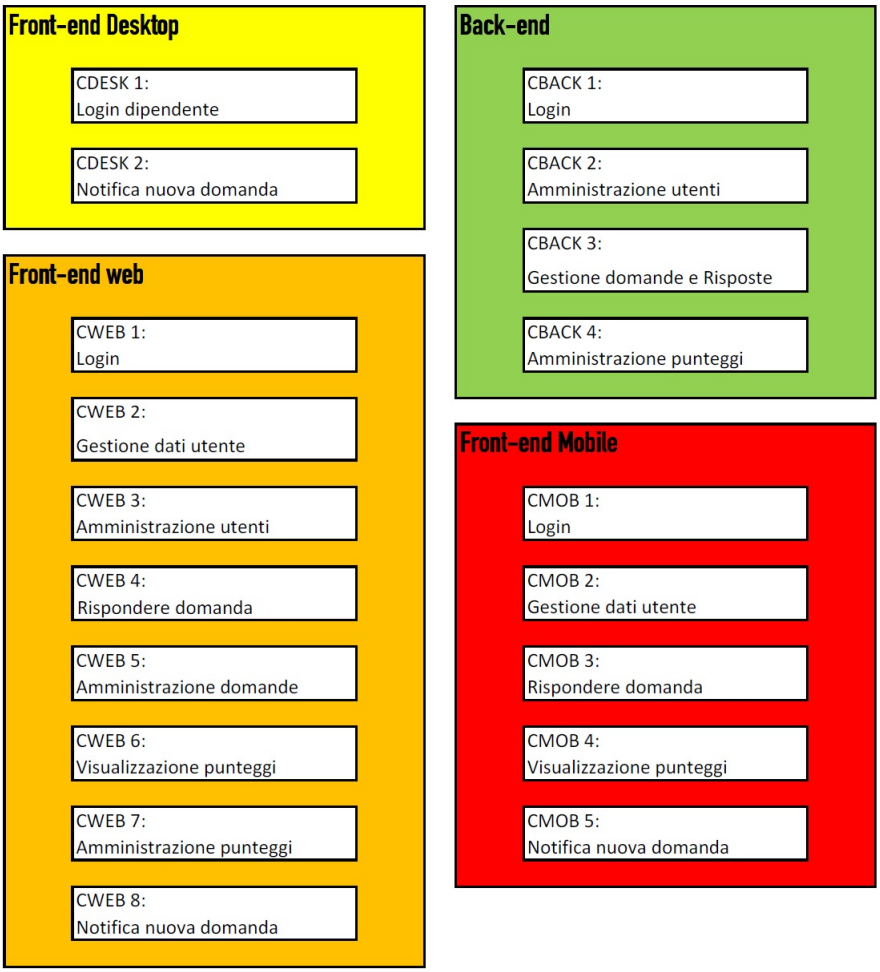

Figura 3: Architettura generale dell'intero sistema SafetyGame

Come specificato nell'*Analisi dei Requisiti*, il front-end è stato diviso in tre parti e principalmente si occuperanno di fornire l'interfaccia grafica agli utenti del sistema, quindi Dipendenti, Amministratori Installatori, Amministratori Sicurezza e Amministratori Azienda. In particolare abbiamo suddiviso il front-end:

- **Desktop:** il "front-end desktop" si occuperà di gestire la temporizzazione delle domande da sottoporre al Dipendente, mostrandogli un pop-up di notifica ogni qualvolta sia giunto il tempo di proporre una nuova domanda. Questo sistema è formato dai seguenti componenti:
  - **CDESK 1:** Login dipendente
  - **CDESK 2:** Notifica nuova domanda
- **Mobile:** il "front-end mobile" si occuperà dei dispositivi mobili Android. É composto dai seguenti componenti:
  - **CMOB 1:** Login
  - **CMOB 2:** Gestione dati utente
  - **CMOB 3:** Rispondere nuova domanda
  - **CMOB 4:** Visualizzazione punteggi
  - **CMOB 5:** Notifica nuova domanda
- **Web:** il "front-end web" gestirà la somministrazione delle domande sia al Dipendente connesso tramite applicazione desktop, sia a tutti gli utenti che sono connessi tramite browser. Esso è composto dai seguenti componenti:
  - **CWEB 1:** Login
  - **CWEB 2:** Gestione dati utente
  - **CWEB 3:** Amministrazione utenti
  - **CWEB 4:** Rispondere nuova domanda
  - **CWEB 5:** Amministrazione domande
  - **CWEB 6:** Visualizzazione punteggi
  - **CWEB 7:** Amministrazione punteggi
  - **CWEB 8:** Notifica nuova domanda

Il back-end si occuperà della persistenza delle informazioni e del loro recupero, in modo da renderle disponibili agli altri sottosistemi. È formato dai seguenti componenti:

- **CBACK 1:** Login
- **CBACK 2:** Amministrazione utenti
- **CBACK 3:** Gestione domande e Risposte
- **CBACK 4:** Amministrazione punteggi

Oltre a questa separazione in componenti, i sottosistemi sono stati divisi in livelli orizzontali, composti da elementi affini tra di loro, come mostrato in figura 4

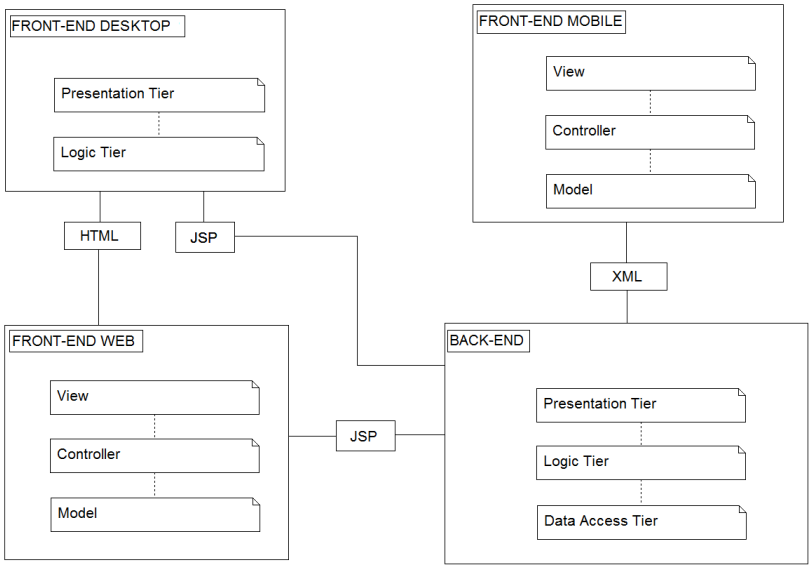

Figura 4: Architettura generale dell'intero sistema

5 Front-end Desktop

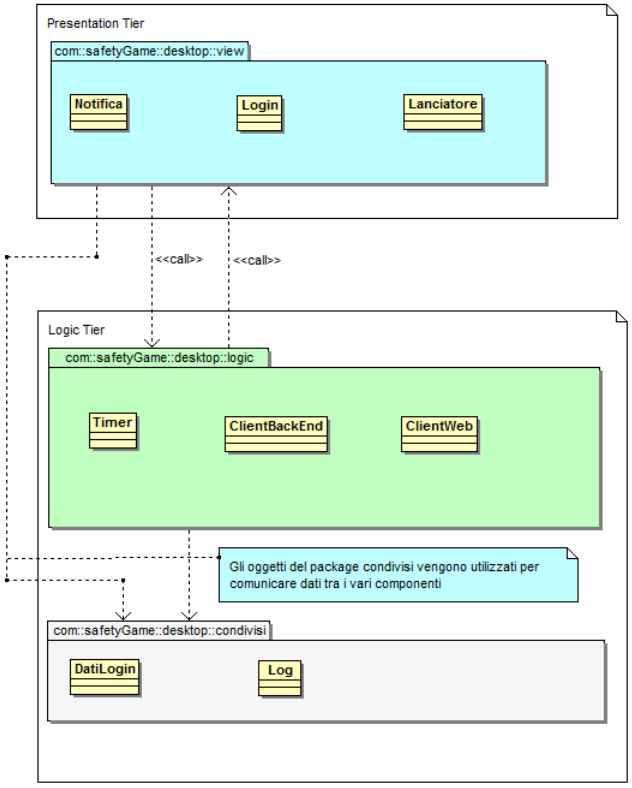

Figure 5: Front-end Desktop

5.1 Presentation Tier

Tipo, obiettivo e funzione del componente: Questo componente costituisce la parte del sistema che definisce i metodi grafici per l'accesso all'applicazione da parte degli utenti e le funzionalità disponibili per un Dipendente Autenticato.
Relazioni d'uso di altre componenti: Questo componente è costituito dal package view.

5.1.1 com.safetyGame.desktop.view.Notifica

Tipo, obiettivo e funzione del componente: La classe Notifica realizza gli oggetti notifiche che compariranno a video al Dipendente Autenticato, le quali gli chiederanno conferma di accettazione o rifiuto per mostrargli la domanda.
Relazioni d'uso di altre componenti: La classe utilizza oggetti di tipo Clien-

tWeb per comunicare con il Dipendente via web nel caso di domanda accettata e oggetti di tipo ClientBackEnd per comunicare con il back-end per aggiornare il log del Dipendente Autenticato.
Interfacce con e relazioni d'uso da altre componenti: Viene richiamato dalla classe Timer e ne fornisce funzioni, richiama le funzioni di ClientBackEnd e ClientWeb.
Attività svolte e dati trattati: Classe puramente grafica, svolge una funzione di presentazione del pop-up che, quando riceve il segnale dalla classe Timer, notifica al Dipendente Autenticato la presenza di una nuova domanda a cui poter rispondere. In seguito resta in attesa dell'input da dipendente sull'accettare la visualizzazione della domanda o sul posticiparla comunicando la relativa risposta al ClientWeb e/o al ClientBackEnd.

5.1.2 com.safetyGame.desktop.view.Login

Tipo, obiettivo e funzione del componente: La classe Login realizza l'oggetto login che serve a gestire il login del Dipendente dalla piattaforma.
Relazioni d'uso di altre componenti: La classe utilizza oggetti di tipo ClientWeb per comunicare al Dipendente i propri dati nel caso di login effettuato con successo e oggetti di tipo ClientBackEnd per comunicare al back end, in caso di successo, di aggiornare il log di quel Dipendente Autenticato, oppure richiedendo una password auto-generata casualmente.
Interfacce con e relazioni d'uso da altre componenti: nessuna.
Attività svolte e dati trattati: Classe puramente grafica, svolge una funzione di presentazione del modulo di login, chiedendo username e password del Dipendente. L'oggetto login viene poi passato al ClientBackEnd stabilire se le credenziali sono valide, in caso affermativo richiama il ClientWeb, in caso negativo ripropone la schermata di login standard. Fornisce inoltre un modulo per la richiesta di generazione casuale della password in caso che questa venga smarrita da parte del Dipendente. La richiesta passa direttamente al ClientBackEnd che la elabora e si rimette in attesa.

5.1.3 com.safetyGame.desktop.view.Lanciatore

Tipo, obiettivo e funzione del componente: La classe Lanciatore realizza l'oggetto lanciatore che crea e gestisce il menù che consente al Dipendente Autenticato di utilizzare le varie funzionalità offerta dal sistema.
Relazioni d'uso di altre componenti: La classe utilizza oggetti di tipo ClientWeb per comunicare le azioni scelte dal Dipendente Autenticato e oggetti di tipo ClientBackEnd per comunicare al back end di aggiornare il log del Dipendente Autenticato con tutte le funzionalità richieste.
Interfacce con e relazioni d'uso da altre componenti: nessuna.
Attività svolte e dati trattati: Classe puramente grafica, svolge una funzione di presentazione del menù (quando richiesto) che consente al Dipendente Autenticato di utilizzare tutte le funzionalità messe a disposizione per lui. Le richieste vengono mandate al ClientWeb che le elabora e la notifica della richi-

esta viene mandata al ClientBackEnd che si preoccuperà principalmente di aggiornare il log del Dipendente Autenticato.

5.2 Logic Tier

Tipo, obiettivo e funzione del componente: Questo componente si occupa di gestire le interazioni che il Dipendente o il Dipendente Autenticato effettua con il sistema attraverso la parte grafica. Tiene traccia dello storico personale di ogni Dipendente Autenticato ed invia dati al back-end per mantenere aggiornato lo stato del sistema.
Relazioni d'uso di altre componenti: Il componente è costituito dal package logic e da quello dati.

5.2.1 com.safetyGame.desktop.logic.Timer

Tipo, obiettivo e funzione del componente: La classe Timer ha l'obiettivo di implementare un contatore temporizzato; al termine del tempo istanzierà un oggetto della classe Notifica.
Relazioni d'uso di altre componenti: La classe utilizza oggetti di tipo Notifica per generare la notifica di una nuova domanda al Dipendente Autenticato.
Interfacce con e relazioni d'uso da altre componenti: Nessuna.
Attività svolte e dati trattati: Decrementerà un contatore con un certo ritardo, in modo da tenere traccia del tempo trascorso dalla sua istanziazione. Allo scadere del tempo, creerà un oggetto di tipo Nofica.

5.2.2 com.safetyGame.desktop.logic.ClientBackEnd

Tipo, obiettivo e funzione del componente: La classe ClientBackEnd, istanzia un oggetto con la funzione di comunicare con la parte back-end. Contiene gli oggetti Log e dati Login utili per la comunicazione con il back-end e con la classe ClientWeb.
Relazioni d'uso di altre componenti: La classe utilizza oggetti di tipo Log e Login per gestire le comunicazioni con i database e con il Presentation-Tier. Scambia l'oggetto Login con la classe ClientWeb. Fornisce le funzioni per la gestione dell'oggetto Log alle classi Login, Lanciatore, Notifica e ClientWeb.
Interfacce con e relazioni d'uso da altre componenti: Ogni classe del Presentation-Tier possiede un riferimento a questa classe, le cui istanze vengono utilizzate per la comunicazione con il database.
Attività svolte e dati trattati: Comunica al back-end gli oggetti di tipo Log di ogni Dipendente Autenticato che verranno in seguito salvati e gli oggetti Login che verranno in seguito controllati. Ritorna il risultato del controllo degli oggetti di tipo Login alla classe ClientWeb.

5.2.3 com.safetyGame.desktop.logic.ClientWeb

Tipo, obiettivo e funzione del componente: La classe ClientWeb, istanzia un oggetto con la funzione di comunicare con la parte web dell'applicazione. Contiene gli oggetti **Login** utili per la comunicazione con la classe ClientBack-End e con il Presentation-Tier.
Relazioni d'uso di altre componenti: La classe utilizza oggetti di tipo Login per gestire le comunicazioni con il Presentation-Tier e con la classe ClientBack-End. Scambia l'oggetto Login con la classe ClientBackEnd e con il Presentation-Tier.
Interfacce con e relazioni d'uso da altre componenti: Ogni classe del Presentation-Tier possiede un riferimento a questa classe, le cui istanze vengono utilizzate per la comunicazione dell'identità del Dipendente e/o del Dipendente Autenticato.
Attività svolte e dati trattati: Comunica alla classe ClientBackEnd l'oggeto di tipo Login per l'autenticazione del Dipendente. Lo stesso oggetto viene poi utilizzato per gestire l'accessibilità delle funzioni del Dipendente Autenticato tra la classe ClientBackEnd e il Presentation-Layer.

5.2.4 com.safetyGame.desktop.condivisi.DatiLogin

Tipo, obiettivo e funzione del componente: La classe DatiLogin istanzia un oggetto che rappresenta le credenziali con cui un Dipendente Autenticato ha effettuato il Login.
Relazioni d'uso di altre componenti: Nessuna.
Interfacce con e relazioni d'uso da altre componenti: La classe DatiLogin viene utilizzata come contenitore dei dati del login del Dipendente. La classe viene utilizzata dalle classi Login e ClientBackEnd e ClientWeb.
Attività svolte e dati trattati: L'oggetto istanziato dalla classe conterrà le credenziali del Login.

5.2.5 com.safetyGame.desktop.condivisi.Log

Tipo, obiettivo e funzione del componente: La classe Log istanzia un oggetto che rappresenta il tracciamento di tutte le azioni effettuate da un Dipendente Autenticato.
Relazioni d'uso di altre componenti: Nessuna.
Interfacce con e relazioni d'uso da altre componenti: La classe Log viene utilizzata come contenitore di dati del tracciamento di tutte le azioni del Dipendente Autenticato. La classe viene utilizzata da tutte le classi del Presentation-Tier e dalle classi ClientBackEnd e ClientWeb.
Attività svolte e dati trattati: L'oggetto istanziato della classe conterrà il tracciamento delle operazioni effettuate da un Dipendente Autenticato.

6 Front-end Web

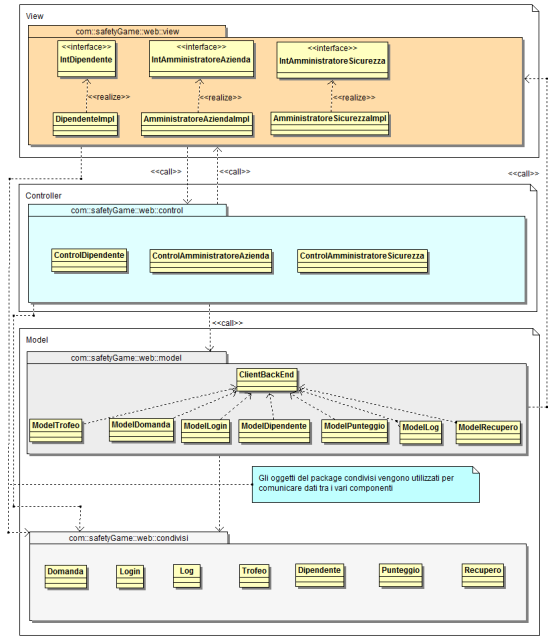

Figure 6: Front-end Web

6.1 View

Tipo, obiettivo e funzione del componente: questo componente costituisce la parte del sistema che definisce ed implementa l'interfaccia web usurfruibile dagli Utenti.
Relazioni d'uso di altre componenti: il componente è costituito dal layer View, e comunica con il layer Model per rendere possibile la memorizzazione delle informazioni immesse nel sistema dagli utenti attraverso l'interfaccia web.

6.1.1 com.safetyGame.web.view.IntDipendente

Tipo, obiettivo e funzione del componente: L'interfaccia IntDipendente fornisce la struttura base per la creazione delle pagine web per i Dipendenti e i Dipendenti Autenticati. Dichiara i metodi comuni ad ogni Dipendente che saranno successivamente implementati.
Relazioni d'uso di altre componenti: Nessuna
Interfacce con e relazioni d'uso da altre componenti: L'interfaccia verrà implementata dalla classe DipendenteImpl.
Attività svolte e dati trattati: L'interfaccia fornisce la base per le implementazioni di tutte le pagine e dei metodi utili per i Dipendenti e i Dipendenti

Autenticati.

6.1.2 com.safetyGame.web.view.IntAmministratoreAzienda

Tipo, obiettivo e funzione del componente: L'interfaccia IntAmministratoreAzienda fornisce la struttura base per la creazione delle pagine web per l'Amministratore Azienda e l'Amministratore Azienda Autenticato. Dichiara i metodi comuni ad ogni Amministratore Azienda che saranno successivamente implementati.
Relazioni d'uso di altre componenti: Nessuna
Interfacce con e relazioni d'uso da altre componenti: L'interfaccia verrà implementata dalla classe AmministratoreAziendaImpl.
Attività svolte e dati trattati: L'interfaccia fornisce la base per le implementazioni di tutte le pagine e dei metodi utili per l'Amministratore Azienda e l'Amministratore Azienda Autenticato.

6.1.3 com.safetyGame.web.view.IntAmministratoreSicurezza

Tipo, obiettivo e funzione del componente: L'interfaccia IntAmministratoreSicurezza fornisce la struttura base per la creazione delle pagine web per l'Amministratore Sicurezza e l'Amministratore Sicurezza Autenticato. Dichiara i metodi comuni ad ogni Amministratore Sicurezza che saranno successivamente implementati.
Relazioni d'uso di altre componenti: Nessuna
Interfacce con e relazioni d'uso da altre componenti: L'interfaccia verrà implementata dalla classe AmministratoreSicurezzaImpl.
Attività svolte e dati trattati: L'interfaccia fornisce la base per le implementazioni di tutte le pagine e dei metodi utili per l'Amministratore Sicurezza e l'Amministratore Sicurezza Autenticato.

6.1.4 com.safetyGame.web.view.DipendenteImpl

Tipo, obiettivo e funzione del componente: La classe DipendenteImpl implementa l'interfaccia IntDipendente definendone tutti i metodi e le pagine web.
Relazioni d'uso di altre componenti: La classe DipendenteImpl implementa metodi e pagine web per la gestione browser del sistema. Utilizza, inoltre, metodi della classe ControlDipendente per verificare i dati inseriti dai Dipendenti e/o Dipendenti Autenticati e scambia con le varie classi gli oggetti del package condivisi.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ControlDipendente per restituire i dati erroneamente modificati e per utilizzarne i metodi di modifica campi dichiarati, ControlAmministratoreAzienda per utilizzare i metodi di modifica campi dichiarati e da tutto il package model che comunica con il back-end.
Attività svolte e dati trattati: La classe implementa l'interfaccia IntDipendente. Richiede la verifica dei dati alla classe ControlDipendente e fornisce

i metodi implementati al package model e alla classe ControlAmministratore-Azienda.

6.1.5 com.safetyGame.web.view.AmministratoreAziendaImpl

Tipo, obiettivo e funzione del componente: La classe Amministratore-AziendaImpl implementa l'interfaccia IntAmministratoreAzienda definendone tutti i metodi e le pagine web.
Relazioni d'uso di altre componenti: La classe AmministratoreAziendaImpl implementa metodi e pagine web per la gestione browser del sistema. Utilizza, inoltre, metodi della classe ControlAmministratoreAzienda per verificare i dati inseriti dagli Amministratori Azienda e/o Amministratori Azienda Autenticati e scambia gli oggetti delle classi Login, Log, Trofeo, Dipendente, Punteggio, Recupero del package condivisi.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ControlAmministratoreAzienda per restituire i dati erroneamente modificati e per utilizzarne i metodi di modifica campi dichiarati. Viene inoltre utilizzata dagli oggetti delle classi ModelTrofeo, ModelLogin, ModelDipendente, ModelPunteggio, ModelLog, ModelRecupero del package model che comunicano con il back-end.
Attività svolte e dati trattati: La classe implementa l'interfaccia IntAmministratoreAzienda. Richiede la verifica dei dati alla classe ControlAmministratore Azienda e fornisce i metodi implementati alle classi sopra elencate del package model.

6.1.6 com.safetyGame.web.view.AmministratoreSicurezzaImpl

Tipo, obiettivo e funzione del componente: La classe AmministratoreSicurezzaImpl implementa l'interfaccia IntAmministratoreSicurezza definendone tutti i metodi e le pagine web.
Relazioni d'uso di altre componenti: La classe AmministratoreSicurezzaImpl implementa metodi e pagine web per la gestione browser del sistema. Utilizza, inoltre, metodi della classe ControlAmministratoreSicurezza per verificare i dati inseriti dagli Amministratori Sicurezza e/o Amministratori Sicurezza Autenticati e scambia gli oggetti delle classi Domanda, Login, Log, Recupero del package condivisi.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ControlAmministratoreSicurezza per restituire i dati erroneamente modificati e per utilizzarne i metodi di modifica campi dichiarati. Viene inoltre utilizzata dagli oggetti delle classi ModelDomanda, ModelLogin, ModelLog, ModelRecupero del package model che comunicano con il back-end.
Attività svolte e dati trattati: La classe implementa l'interfaccia IntAmministratoreSicurezza. Richiede la verifica dei dati alla classe ControlAmministratore Sicurezza e fornisce i metodi implementati alle classi sopra elencate del package model.

6.2 Controller

Tipo, obiettivo e funzione del componente: Questo livello si occupa di controllare la sussistenza dei dati da e verso l'interfaccia grafica visibile all'utente. È incaricato di mantenere aggiornati i dati visualizzati dall'interfaccia grafica web interrogando quelli presenti nel sistema.
Relazioni d'uso di altre componenti: Il livello è costituito dal package control. Comunica con il livello Model per accedere ai dati presenti nel sistema, e con il livello View per aggiornare la parte grafica web.
inserire frase per controlli identità coerente

6.2.1 com.safetyGame.web.control.ControlDipendente

Tipo, obiettivo e funzione del componente: La classe ControlDipendente si interpone tra lo scambio di dati fra il back-end ed gli oggetti Dipendente istanziati_in_DipendenteImpl. Esegue i controlli per verificare che l'identità che viene passata dal DipendenteImpl sia coerente con i dati presenti nel back-end.
Relazioni d'uso di altre componenti: La classe ControlDipendente utilizza le implementazioni della classe DipendenteImpl e comunica con il back-end tramite il package model.
Interfacce con e relazioni d'uso da altre componenti: La classe viene usata da DipendenteImpl per effettuare i controlli sui vari input che un Dipendente o un Dipendente Autenticato immette.
Attività svolte e dati trattati: La classe esegue i controlli di consistenza di tutti i dati immessi nell'oggetto istanziato dalla classe DipendenteImpl verso il Back-end.

6.2.2 com.safetyGame.web.control.ControlAmministratoreAzienda

Tipo, obiettivo e funzione del componente: La classe ControlAmministratoreAzienda si interpone tra lo scambio di dati fra il back-end ed gli oggetti Amministratore_Azienda istanziati_in_AmministratoreAziendaImpl. Esegue i controlli per verificare che l'identità che viene passata dal AmministratoreAziendaImpl sia coerente con i dati presenti nel back-end.
Relazioni d'uso di altre componenti: La classe ControlAmministratoreAzienda utilizza le implementazioni della classe AmministratoreAziendaImpl e comunica con il back-end tramite il package model.
Interfacce con e relazioni d'uso da altre componenti: La classe viene usata da AmministratoreAziendaImpl per effettuare i controlli sui vari input che un Amministratore Azienda o un Amministratore Azienda Autenticato immette.
Attività svolte e dati trattati: La classe esegue i controlli di consistenza di tutti i dati immessi nell'oggetto istanziato dalla classe AmministratoreAziendaImpl verso il Back-end.

6.2.3 com.safetyGame.web.control.ControlAmministratoreSicurezza

Tipo, obiettivo e funzione del componente: La classe ControlAmministratoreSicurezza si interpone tra lo scambio di dati fra il back-end ed gli oggetti Amministratore Sicurezza istanziati in AmministratoreSicurezzaImpl. Esegue i controlli per verificare che l'identità che viene passata dal AmministratoreSicurezzaImpl sia coerente con i dati presenti nel back-end.
Relazioni d'uso di altre componenti: La classe ControlAmministratoreSicurezza utilizza le implementazioni della classe AmministratoreSicurezzaImpl e comunica con il back-end tramite il package model.
Interfacce con e relazioni d'uso da altre componenti: La classe viene usata da AmministratoreSicurezzaImpl per effettuare i controlli sui vari input che un Amministratore Sicurezza o un Amministratore Sicurezza Autenticato immette.
Attività svolte e dati trattati: La classe esegue i controlli di consistenza di tutti i dati immessi nell'oggetto istanziato dalla classe AmministratoreSicurezzaImpl verso il Back-end.

6.3 Model

Tipo, obiettivo e funzione del componente: Questo livello contiene e rappresenta i dati veri e propri del sistema. Offre le classi che consentono di interfacciarsi ai dati contenuti nel back-end del sistema.
Relazioni d'uso di altre componenti: Il Model utilizza le funzioni del View per visualizzare i risultati delle azioni.
Interfacce con e relazioni d'uso da altre componenti: Le funzioni di Model sono utilizzate da Controller.

6.3.1 com.safetyGame.web.model.ClientBackEnd

Tipo, obiettivo e funzione del componente: La classe ClientBackEnd viene utilizzata per effettuare la connessione fisica tra il Front-end Web ed il Back-end.
Relazioni d'uso di altre componenti: Richiama metodi del Presentation Tier del Back-end in base alle richieste delle classi del Model.
Interfacce con e relazioni d'uso da altre componenti: Viene richiamata dalle classi ModelXxx per inviare o ricevere dati al Back-end.
Attività svolte e dati trattati: La classe si occupa di effettuare le richieste e ricevere risposte utilizzando script JSP.

6.3.2 com.safetyGame.web.model.ModelXxx

Tipo, obiettivo e funzione del componente: Le classi ModelXxx vengono utilizzate per inviare e richiedere dati al Back-end attraverso la classe ClientBackEnd.
Relazioni d'uso di altre componenti: Utilizzano metodi del componente

View per visualizzare i dati ottenuti, utilizzano la classe ClientBackEnd per comunicare col server; Utilizzano oggetti del package condivisi per scambiare dati con gli altri componenti.
Interfacce con e relazioni d'uso da altre componenti: Vengono richiamate dalle classi di Controller per inviare dati al server, vengono richiamate dalle classi di View per far visualizzare dati presenti nel server.
Attività svolte e dati trattati: Le classi ricevono richieste di scambio dati col database dalle relative classi di Controller, utilizzano la classe ClientBackEnd per comunicare con le classi relative nel Presentation Tier del Back-end.

6.3.3 com.safetyGame.web.condivisi.Xxx

Tipo, obiettivo e funzione del componente: Le classi Xxx vengono utilizzate come contenitori per trasferire dati tra i vari componenti.
Relazioni d'uso di altre componenti: Nessuna.
Interfacce con e relazioni d'uso da altre componenti: Le classi vengono utilizzate da tutti gli altri package per trasferire informazioni tra i vari livelli del Front-end.
Attività svolte e dati trattati: Ogni classe contiene metodi per leggere, inserire o modificare i dati contenuti al suo interno.

7 Front-end Mobile

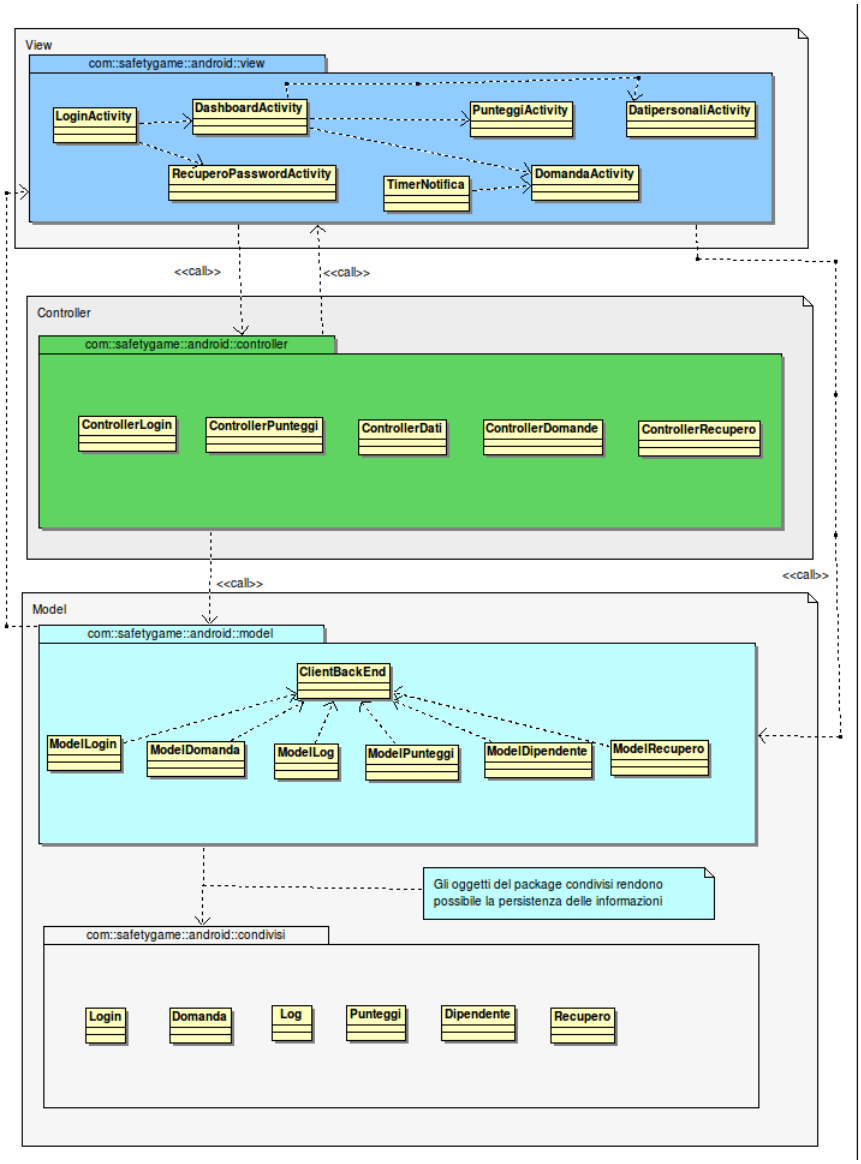

Figure 7: Front-end Mobile

7.1 View

Tipo, obiettivo e funzione del componente: Contiene tutte le classi Java che estendono la classe Java di Android "Activity" e che quindi riguardano la

grafica dell'applicazione. Anche la classe TimerNotifica fa parte di questo componente anche se estende la classe Java Android "Service".
Relazioni d'uso di altre componenti: Utilizza funzioni del Controller per il controllo ed invio dei dati, utilizza il Model per ottenere dati dal server.
Interfacce con e relazioni d'uso da altre componenti: View può venire modificata da Controller.

7.1.1 com.safetyGame.android.view.LoginActivity:

Tipo, obiettivo e funzione del componente: Activity che permette di effettuare il login ad un Dipendente**.**
Relazioni d'uso di altre componenti: La classe richiama ControllerLogin per inviare i dati al server, inoltre consente di arrivare alle classi DashboardActivity e RecuperoPasswordActivity; Utilizza oggetti di tipo Login per trasferire i dati.
Interfacce con e relazioni d'uso da altre componenti: Riceve la risposta tramite una chiamata di metodo da ModelLogin.
Attività svolte e dati trattati: Raccoglie i dati inseriti dal Dipendente Autenticato e invia la richiesta di login al Back-end.

7.1.2 com.safetyGame.android.view.DashboardActivity:

Tipo, obiettivo e funzione del componente: Activity che permette di raggiungere le altre Activity del programma.
Relazioni d'uso di altre componenti: Può chiamare le classi PunteggiActivity, DomandaActivity e DatipersonaliActivity.
Interfacce con e relazioni d'uso da altre componenti: Viene chiamato da LoginActivity.
Attività svolte e dati trattati: Intercetta gli input e lancia altre Activity.

7.1.3 com.safetyGame.android.view.PunteggiActivity:

Tipo, obiettivo e funzione del componente: Activity che permette ad un Dipendente Autenticato di visualizzare i vari punteggi e trofei.
Relazioni d'uso di altre componenti: Utilizza ControllerPunteggi per richiedere dati al server, utilizza oggetti di tipo Punteggi per trasferire dati.
Interfacce con e relazioni d'uso da altre componenti: Viene chiamato da DashboardActivity e riceve i dati da ModelPunteggi.
Attività svolte e dati trattati: Visualizza i punteggi e trofei del Dipendente Autenticato.

7.1.4 com.safetyGame.android.view.DatipersonaliActivity:

Tipo, obiettivo e funzione del componente: Activity che permette ad un Dipendente Autenticato di visualizzare e modificare i propri dati personali.
Relazioni d'uso di altre componenti: Utilizza ControllerDati per richiedere

ed inviare modifiche riguardanti dati personali del Dipendente Autenticato, utilizza oggetti di tipo Dipendente per trasferire i dati.
Interfacce con e relazioni d'uso da altre componenti: È chiamato da DashboardActivity e riceve i dati richiesti da ModelDati.
Attività svolte e dati trattati: Visualizza e permette di modificare i dati personali del Dipendente Autenticato.

7.1.5 com.safetyGame.android.view.DomandaActivity:

Tipo, obiettivo e funzione del componente: Activity che permette ad un Dipendente Autenticato di visualizzare, richiedere e risponde alle domande.
Relazioni d'uso di altre componenti: utilizza ControllerDomanda per richiedere dati al server, utilizza oggetti di tipo Domanda per trasferire dati.
Interfacce con e relazioni d'uso da altre componenti: È chiamato da DashboardActivity o TimerNotifica e riceve i dati da ModelDati.
Attività svolte e dati trattati: Permette di richiedere, visualizzare e rispondere a domande.

7.1.6 com.safetyGame.android.view.RecuperoPasswordActivity:

Tipo, obiettivo e funzione del componente: Activity che permette di recuperare la password di un Dipendente.
Relazioni d'uso di altre componenti: utilizza ControllerRecuperoper l'invio di dati al server.
Interfacce con e relazioni d'uso da altre componenti: È chiamato da LoginActivity e riceve dati da ModelRecupero.
Attività svolte e dati trattati: Raccoglie i dati del Dipendente e richiede il recupero password.

7.1.7 com.safetyGame.android.view.TimerNotifica:

Tipo, obiettivo e funzione del componente: Servizio timer che proporrà al Dipendente Autenticato una domanda secondo le specifiche impostate attraverso le notifiche standard di Android.
Relazioni d'uso di altre componenti: Chiama la classe DomandaActivity.
Interfacce con e relazioni d'uso da altre componenti: Nessuna.
Attività svolte e dati trattati: Periodicamente un timer a seconda delle impostazioni farà partire una notifica standard di Android che permette di visualizzare una nuova domanda.

7.2 Controller

Tipo, obiettivo e funzione del componente: Contiene le classi di utilità che raggruppano i metodi chiamati al compimento delle azioni in input dalla View i quali controllano la correttezza degli input e chiamano i giusti metodi del Model o chiamano altri metodi della View.
Relazioni d'uso di altre componenti: Utilizza metodi sia di Model che di

View.
Interfacce con e relazioni d'uso da altre componenti: Viene utilizzato da View.

7.2.1 com.safetyGame.android.control.ControllerXxx

Tipo, obiettivo e funzione del componente: Le classi ControllerXxx forniscono metodi ad ogni Activity relativa per gestire gli input del Dipendente Autenticato.
Relazioni d'uso di altre componenti: Richiamano metodi del Model per ottenere i dati richiesti dal Dipendente Autenticato, in caso di fallimento, richiamano metodi di View per la visualizzazione dei messaggi d'errore; Utilizzano oggetti del package condivisi per scambiare i dati con gli altri componenti.
Interfacce con e relazioni d'uso da altre componenti: Vengono chiamate dalle rispettive classi dal componente View.
Attività svolte e dati trattati: Ad ogni input nelle classi di View vengono chiamate le relative classi del Controller, le quali gestiranno e controlleranno i dati in input ed in seguito chiameranno i giusti metodi del componente Model od in caso, di View.

7.3 Model

Tipo, obiettivo e funzione del componente: Contiene le classi che comunicano con le API del server, ne ricevono i dati e li gestiscono.
Relazioni d'uso di altre componenti: Comunica con il Presentation Tier del Back-end e notifica al componente View quando i dati sono pronti.
Interfacce con e relazioni d'uso da altre componenti: È utilizzato dal componente Controller.
Attività svolte e dati trattati: Invia delle richieste HTTP al server, il quale gli risponderà inviando i dati richiesti attraverso un XML, in seguito ne estrapolerà i dati e li renderà disponibili alla View.

7.3.1 com.safetyGame.android.model.ClientBackEnd

Tipo, obiettivo e funzione del componente: La classe ClientBackEnd viene utilizzata per effettuare la connessione fisica tra il Front-end Mobile ed il Back-end.
Relazioni d'uso di altre componenti: Richiama metodi del Presentation Tier del Back-end in base alle richieste delle classi del Model.
Interfacce con e relazioni d'uso da altre componenti: Viene richiamata dalle classi ModelXxx per inviare o ricevere dati al Back-end.
Attività svolte e dati trattati: La classe si occupa di effettuare le richieste HTTP al server e di riceverne le risposte tramite pagine XML.

7.3.2 com.safetyGame.android.model.ModelXxx

Tipo, obiettivo e funzione del componente: Le classi ModelXxx vengono utilizzate per inviare e richiedere dati al Back-end attraverso la classe ClientBackEnd.
Relazioni d'uso di altre componenti: Utilizzano metodi del componente View per visualizzare i dati ottenuti, utilizzano la classe ClientBackEnd per comunicare col server; Utilizzano oggetti del package condivisi per scambiare dati con gli altri componenti.
Interfacce con e relazioni d'uso da altre componenti: Vengono richiamate dalle classi ControllerXxx per inviare dati al server, vengono richiamate dalle classi di View per far visualizzare dati presenti nel server.
Attività svolte e dati trattati: Le classi ricevono richieste di scambio dati col database dalle relative classi ControllerXxx, utilizzano la classe ClientBackEnd per comunicare con le classi relative nel Presentation Tier del Back-end.

7.3.3 com.safetyGame.android.condivisi.Xxx

Tipo, obiettivo e funzione del componente: Le classi Xxx vengono utilizzate come contenitori per trasferire dati tra i vari componenti.
Relazioni d'uso di altre componenti: Nessuna.
Interfacce con e relazioni d'uso da altre componenti: Le classi vengono utilizzate da tutti gli altri package per trasferire informazioni tra i vari livelli del Front-end.
Attività svolte e dati trattati: Ogni classe contiene metodi per leggere, inserire o modificare i dati contenuti al suo interno.

8 Back-end

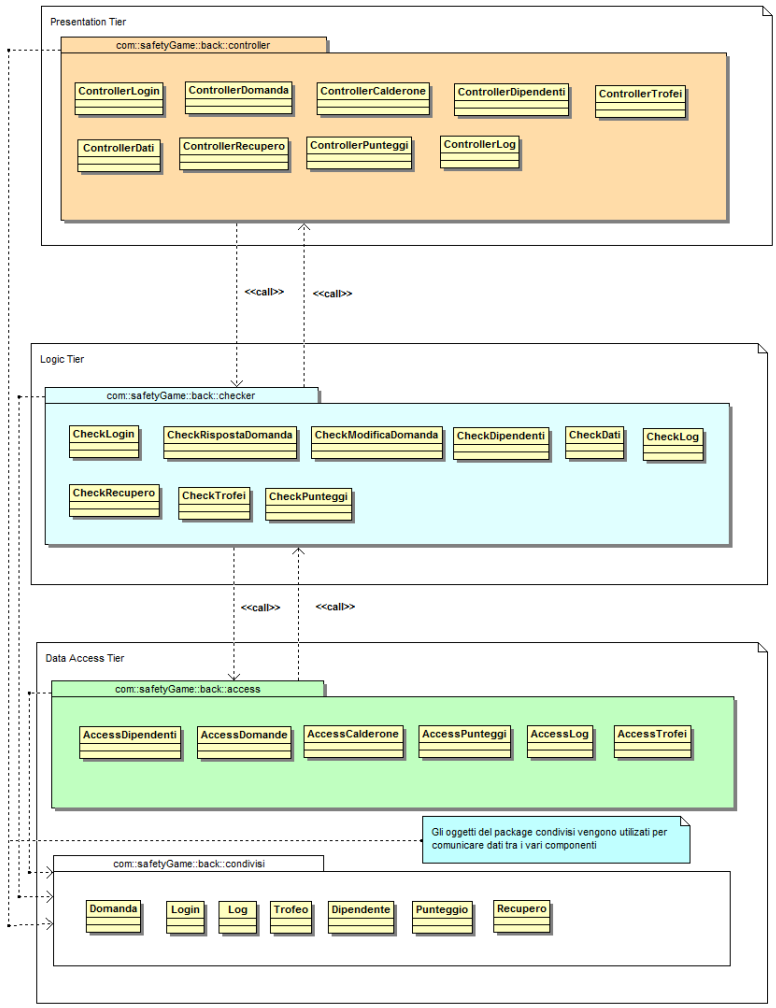

Figure 8: Back-end

8.1 Presentation Tier

Tipo, obiettivo e funzione del componente: Il Presentation Tier si occupa di gestire le richieste degli applicativi Front-end, inviandole alle corrette funzioni del Logic Tier ed inviando i risultati di tali operazioni ai relativi Front-end.
Relazioni d'uso di altre componenti: Il Presentation Tier richiama funzioni dal Logic Tier e dalle classi ClientBackEnd dei vari Front-end.

8.1.1 com.safetyGame.back.controller.ControllerLogin

Tipo, obiettivo e funzione del componente: La classeControllerLogin viene utilizzata per fornire funzioni di login agli applicativi Front-end.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckLogin del package Checker ed utilizza oggetti di tipo Login del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ClientBackEnd dei vari Front-end.
Attività svolte e dati trattati: La classe riceve richieste di autenticazione tramite la ricezione di oggetti Login e le invia a Checklogin per verificarne la consistenza, notificando il ClientBackEnd del successo o fallimento dell'operazione.

8.1.2 com.safetyGame.back.controller.ControllerDomanda

Tipo, obiettivo e funzione del componente: La classe ControllerDomanda viene utilizzata per gestire richieste di visualizzazione di domande e di invio risposte da parte dei componenti Front-end.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckRispostaDomanda del package checker ed utilizza oggetti di tipo Domanda del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ClientBackEnd dei vari Front-end e provvede metodi a CheckRispostaDomanda per inviare i dati ottenuti ai relativi Front-end.
Attività svolte e dati trattati: La classe riceve richieste di visualizzazione o risposta di una domanda da parte di un ClientBackEnd, richiamando i metodi di CheckRispostaDomanda per ottenere una domanda da visualizzare o verificare la risposta data.

8.1.3 com.safetyGame.back.controller.ControllerCalderone

Tipo, obiettivo e funzione del componente: La classe ControllerCalderone viene utilizzata per gestire richieste di visualizzazione, aggiunta ed eliminazione di domande nel database locale di domande, inoltre viene utilizzata per richieste di visualizzazione di domande presenti nel database centrale di domande.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckModificaDomanda del package checker ed utilizza oggetti di tipo Domanda del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalla classe ClientBackEnd del Front-end web e provvede metodi a CheckModificaDomanda per inviare i dati ottenuti al Front-end.
Attività svolte e dati trattati: La classe riceve richieste di visualizzazione, aggiunta od eliminazione di una o più domande dal database locale di domande o di visualizzazione di domande dal database centrale di domande e richiama i relativi metodi di CheckModificaDomanda per effettuare le operazioni richieste.

8.1.4 com.safetyGame.back.controller.ControllerDipendenti

Tipo, obiettivo e funzione del componente: La classe ControllerDipendenti viene utilizzata per gestire richieste di visualizzazione, aggiunta, modifica ed eliminazione di account Dipendente dal database aziendale di dipendenti.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckDipendenti del package checker ed utilizza oggetti di tipo Dipendente del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalla classe ClientBackEnd del Front-end web e provvede metodi a CheckDipendenti per inviare i dati ottenuti al Front-end.
Attività svolte e dati trattati: La classe riceve richieste di visualizzazione, aggiunta, modifica od eliminazione di account Dipendente e richiama i relativi metodi di CheckDipendente per controllare la consistenza ed effettuare le operazioni richieste.

8.1.5 com.safetyGame.back.controller.ControllerTrofei

Tipo, obiettivo e funzione del componente: La classe ControllerTrofei viene utilizzata per gestire richieste di visualizzazione, aggiunta, modifica ed eliminazione di Trofei dal database aziendale di trofei.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckTrofei del package checker ed utilizza oggetti di tipo Trofeo del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalla classe ClientBackEnd del Front-end web per aggiunta, modifica ed eliminazione di trofei, mentre viene utilizzata da tutti i Front-end per la visualizzazione di trofei; Provvede metodi a CheckTrofei per inviare i dati ottenuti ai relativi Front-end.
Attività svolte e dati trattati: La classe riceve richieste di visualizzazione, aggiunta, modifica od eliminazione di trofei e richiama i relativi metodi di CheckTrofei per controllare la consistenza ed effettuare le operazioni richieste.

8.1.6 com.safetyGame.back.controller.ControllerDati

Tipo, obiettivo e funzione del componente: La classe ControllerDati viene utilizzata per gestire richieste di visualizzazione e modifica di dati account Dipendente dal database aziendale di dipendenti o per gestire richieste di modifica password di account Amministratore Azienda.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckDati del package checker ed utilizza oggetti di tipo Dipendente del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ClientBackEnd dei Front-end per i dati account Dipendente e dal ClientBackEnd web per la password Amministratore Azienda; Provvede metodi a CheckDipendenti per inviare i dati ottenuti ai relativi Front-end.
Attività svolte e dati trattati: La classe riceve richieste di visualizzazione o

modifica di dati account Dipendente o di modifica password di account Amministratore Azienda e richiama i relativi metodi di CheckDati per controllare la consistenza ed effettuare le operazioni richieste.

8.1.7 com.safetyGame.back.controller.ControllerRecupero

Tipo, obiettivo e funzione del componente: La classe ControllerRecupero viene utilizzata per gestire richieste di recupero password per account Dipendente, Amministratore Azienda ed Amministratore Sicurezza.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckRecupero del package checker ed utilizza oggetti di tipo Recupero del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ClientBackEnd dei Front-end web e mobile, provvede metodi a CheckRecupero per inviare i dati ottenuti ai relativi Front-end.
Attività svolte e dati trattati: La classe riceve richieste di recupero password di account Dipendente, Amministratore Azienda ed Amministratore Sicurezza e richiama i relativi metodi di CheckRecupero per controllare la consistenza ed effettuare le operazioni richieste.

8.1.8 com.safetyGame.back.controller.ControllerPunteggi

Tipo, obiettivo e funzione del componente: La classe ControllerPunteggi viene utilizzata per gestire richieste di visualizzazione di punteggi relativi ad account Dipendente dal database aziendale di dipendenti.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckPunteggi del package checker ed utilizza oggetti di tipo Punteggio del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ClientBackEnd dei Front-end e provvede metodi a CheckPunteggi per inviare i dati ottenuti ai Front-end.
Attività svolte e dati trattati: La classe riceve richieste di visualizzazione di punteggi relativi ad account Dipendente e richiama i relativi metodi di CheckPunteggi per controllare la consistenza ed effettuare le operazioni richieste.

8.1.9 com.safetyGame.back.controller.ControllerLog

Tipo, obiettivo e funzione del componente: La classe ControllerLog viene utilizzata per gestire richieste di aggiunta di Log per il tracciamento delle azioni dei Dipendenti all'interno del sistema.
Relazioni d'uso di altre componenti: La classe utilizza metodi della classe CheckLog del package checker ed utilizza oggetti di tipo Log del package condivisi per scambiare dati.
Interfacce con e relazioni d'uso da altre componenti: La classe viene utilizzata dalle classi ClientBackEnd dei Front-end.
Attività svolte e dati trattati: La classe riceve richieste di aggiunta di Log

relativi ad account Dipendente e richiama i relativi metodi di CheckLog per controllare la consistenza ed effettuare le operazioni richieste.

8.2 Logic Tier

Tipo, obiettivo e funzione del componente: Il Logic Tier si occupa di far scambiare gli oggetti contenenti dati tra il Presentation Tier ed il Data Access Tier, controllandone la conformità dei dati contenuti.
Relazioni d'uso di altre componenti: Il Logic Tier richiama funzioni sia dal Presentation Tier che dal Data Access Tier.

8.2.1 com.safetyGame.back.checker.CheckXxx

Tipo, obiettivo e funzione del componente: Le classi CheckXxx vengono utilizzate per il controllo ed il trasferimento dei dati richiesti dal Presentation Tier e dal Data Access Tier**.**
Relazioni d'uso di altre componenti: Le classi CheckXxx utilizzano metodi delle classi del package access per ottenere ed inviare i dati richiesti dal package controller ed utilizzano metodi delle classi controller per inviare i dati ottenuti dal package access a chi li ha richiesti; utilizzano oggetti del package condivisi per scambiare dati con gli altri due package.
Interfacce con e relazioni d'uso da altre componenti: Le classi CheckXxx vengono richiamate dalle relative classi controller per ricevere le richieste degli utenti e dalle classi access per effettuare il passaggio di dati.
Attività svolte e dati trattati: Le classi CheckXxx forniscono metodi alle relative classi dei package controller e access per far transitare oggetti contenenti dati dal Presentation Tier al Data Access Tier e viceversa, effettuando controlli di conformità dei dati contenuti in tali oggetti.

8.3 Data Access Tier

Tipo, obiettivo e funzione del componente: Il Data Access Tier si occupa di accedere ai vari database e di creare oggetti del package condivisi per trasferire i dati ottenuti ai livelli superiori.
Relazioni d'uso di altre componenti: Il Data Access Tier richiama funzioni dal Logic Tier.

8.3.1 com.safetyGame.back.access.AccessXxx

Tipo, obiettivo e funzione del componente: Le classi AccessXxx vengono utilizzate per interrogare le tabelle dei database secondo le richieste del Logic Tier.
Relazioni d'uso di altre componenti: Le classi utilizzano metodi del package checker per inviare dati al Presentation Tier ed usa query per interrogare i database**.**
Interfacce con e relazioni d'uso da altre componenti: Le classi vengono

utilizzate dalle relative classi del package checker per cercare o modificare dati presenti nei database.
Attività svolte e dati trattati: Le classi ricevono richieste dalle classi del package cecker riguardanti interrogazioni del database, una volta ottenuti i dati richiesti li inviano tramite metodi del package checker ai livelli più alti.

8.3.2 com.safetyGame.back.condivisi.Xxx

Tipo, obiettivo e funzione del componente: Le classi Xxx vengono utilizzate come contenitori per trasferire dati tra i vari Tier.
Relazioni d'uso di altre componenti: Nessuna.
Interfacce con e relazioni d'uso da altre componenti: Le classi vengono utilizzate da tutti gli altri package per trasferire informazioni tra i vari livelli del Back-end.
Attività svolte e dati trattati: Ogni classe contiene metodi per leggere, inserire o modificare i dati contenuti al suo interno.

9 Diagrammi di attività

In questa sezione si illustreranno diagrammi di attività relativi alle possibili interazioni di un utente con il front-end, suddividendo tali attivita nei tre ambiti di Dipendente, Amministratore Sicurezza e Amministratore Azienda. Tutti e tre gli utenti possono chiedere la modifica da parte del sistema della propria password prima di accedere al sistema, il quale ne genererà una e la renderà nota solo a quell'utente; oppure possono effettuare il login all'interno del sistema e utilizzare quello che il sistema gli offre.

9.1 Ambito Dipendente

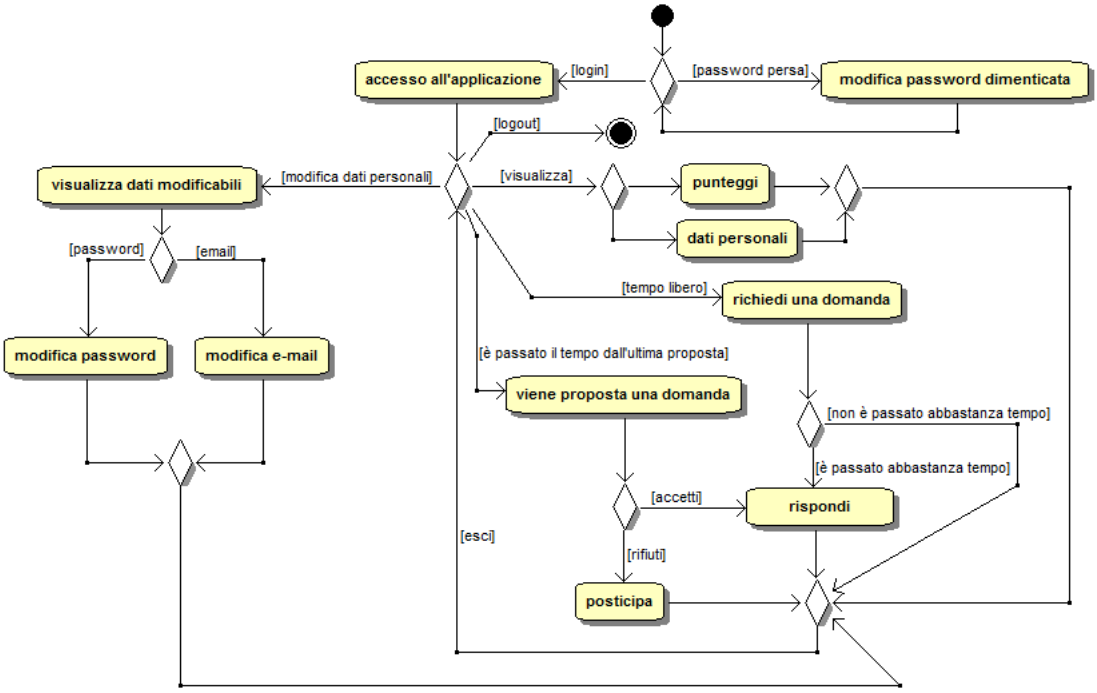

Figura 9: Ambito Dipendente

Il Dipendente, dopo che ha effettuato il login, si trova a disposizione diverse opportunità di azioni.

L'azione base che può fare è semplicemente aspettare che il sistema gli proponga di rispondere ad una domanda, alla quale può decidere se rifiutare (posticipandola) oppure accettare e quindi facendosela mostrare. L'azione di mostrare una domanda obbliga il Dipendente a rispondere alla domanda, nel caso che il Dipendente non sappia la risposta, può dare risposta nulla,

ma deve in ogni caso rispondere. L'applicazione che fa vedere questa domanda (browser o applicazione Android) impedirà la chiusura da parte dell'utente dell'applicazione.

È prevista una alternativa all'attesa di proposta domanda, ovvero la richiesta di una domanda, alla quale il sistema reagirà controllando il tempo trascorso dall'ultima domanda richiesta. Se il lasso di tempo trascorso si rivelerà essere sufficientemente grande, allora all'utente verrà mostrata la domanda alla quale si applicano le stesse regole riportate nel caso precedente. Nel caso in cui, invece, il tempo trascorso non fosse sufficiente, il sistema risponderà con un messaggio d'allerta e ritornerà alla pagina principale delle azioni.

Il Dipendente può inoltre modificare il proprio recapito e-mail e la propria password di autenticazione. Una volta che una di queste due azioni è stata effettuata, il sistema si riporterà sulla schermata principale.

L'ultimo insieme di azioni che può fare il Dipendente è la visualizzazione dei propri punteggi ottenuti durante l'utilizzo del sistema e i propri dati personali salvati in esso.

Infine può effettuare il logout, il quale consentirà al Dipendente di uscire dalla sessione autenticata, salvando ogni azione che è stata effettuata, senza però permettere di chiudere qualsiasi applicazione (nel caso si tratti di una applicazione per dispositivi fissi o dispositivi mobili).

9.2 Ambito Amministratore Azienda

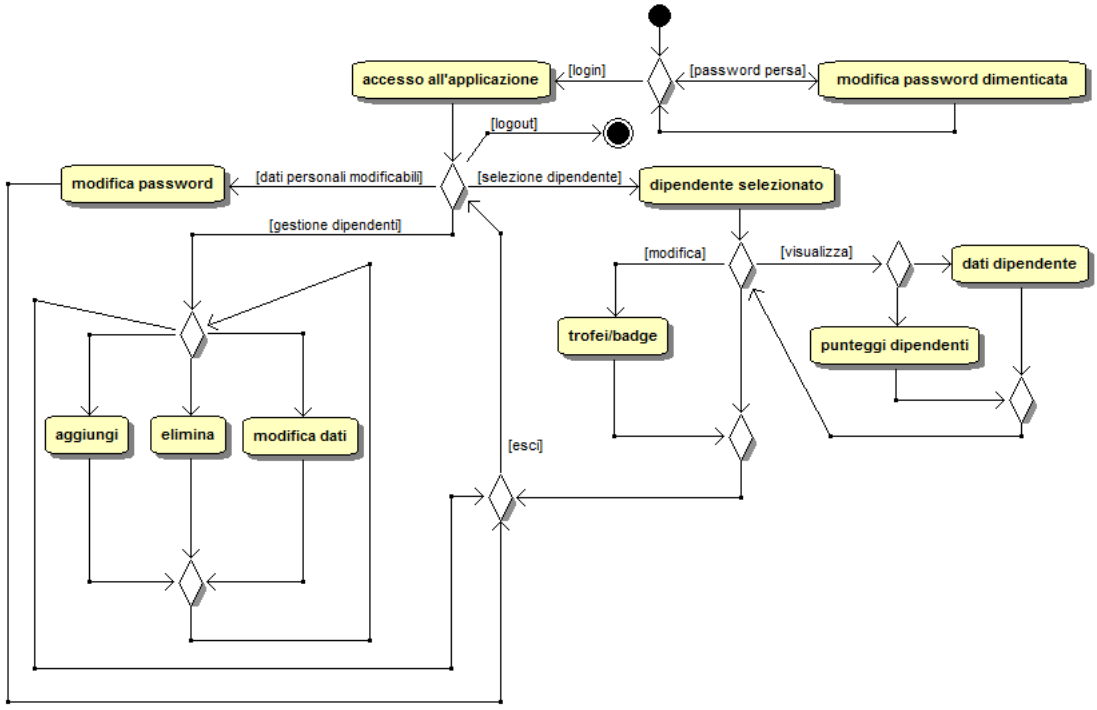

Figura 10: Ambito Amministratore Azienda

L'Amministratore Azienda, dopo che ha effettuato il login, si trova a disposizione diverse opportunità di azioni.

Come azione principale, l'Amministratore Azienda può gestire i dipendenti. La gestione dei Dipendenti si intende una scelta tra l'aggiunta di un account Dipendente al sistema, la cancellazione di un account Dipendente dal sistema (e di tutte le informazioni ad esso associate) e infine alla modifica dei dati personali del Dipendente.

- **L'aggiunta di un account** Dipendente richiede dei campi personali in modo da identificare univocamente una persona, in seguito username e password saranno associate automaticamente dal sistema e, sempre automaticamente, inviate alla e-mail specificata
- **La modifica dei dati** dell'account di un Dipendente, permette all'Amministratore Azienda di modificare, in casi eccezionali, i dati che sono stati messi nel sistema per identificare quella persona

Oltre alla modifica dei dati personali dei Dipendenti, è prevista la possibilità di modificare la propria password di autenticazione ricevendone conferma nell'indirizzo specificato.

L'ultimo insieme di azioni che può fare l'Amministratore Azienda, dopo aver selezionato un Dipendente, è la visualizzazione dei punteggi ottenuti durante l'utilizzo del sistema, i dati personali del Dipendente e infine la possibilità di aggiungere o rimuovere manualmente trofei/badge.

Infine può effettuare il logout, il quale consentirà al Amministratore Azienda di uscire dalla sessione autenticata, salvando ogni azione che è stata effettuata.

9.3 Ambito Amministratore Sicurezza

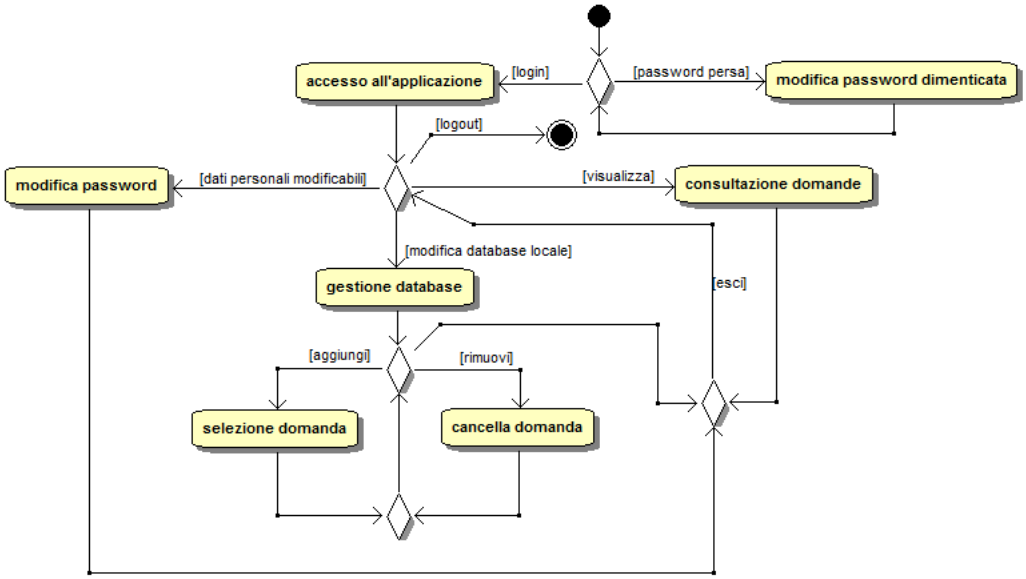

Figura 11: Ambito Amministratore Sicurezza

L'Amministratore Sicurezza, dopo che ha effettuato il login, si trova a disposizione diverse opportunità di azioni.

Come azione principale, l'Amministratore Sicurezza può gestire il database di domande interno dell'azienda. La gestione database di domande viene effettuata dal sito dell'azienda da parte di un unico Amministratore Sicurezza, il quale può:

- **aggiungere** una domanda all'insieme di domande interno (prese da un database generale esterno)
- **cancellare** una domanda, sempre dal database locale

Oltre alla modifica database di domande, è prevista la possibilità consultare l'elenco in dettaglio delle domande locali.

Può inoltre modificare la propria password personale per l'accesso a quell'azienda. Una volta modificata la password, viene inviata una notifica all'indirizzo mail dell'Amministratore Sicurezza.

Infine può effettuare il logout, il quale consentirà al Amministratore Sicurezza di uscire dalla sessione autenticata, salvando ogni azione che è stata effettuata.

9.4 Considerazioni finali

Gli ambiti dell'Amministratore Azienda e Amministraotre Sicurezza, verranno gestiti unicamente tramite web.

L'Ambito Dipendente invece, verrà gestito tramite web, applicazione Android e applicazione per dispositivi fissi. Tutte e tre le applicazioni permettono di eseguire ogni attività illustrata, tuttavia, l'applicazione per dispositivi fissi fornirà un comodo menù per ogni azione (una volta effettuato il login); il quale, una volta attivata la corrispondente voce nel menù, aprirà il browser e quindi sfrutterà la parte web del sistema. Invece l'applicazione Android sfrutterà le peculiarità del sistema stesso per gestire ogni singolo aspetto delle azioni intraprendibili dal Dipendente.

10 Tracciamento componenti - requisiti

10.1 Desktop

Package	Classe	Requisiti
view	Notifica	RFOB 1 RFOB 6 RFOB 6.1 RFOB 13 RQOP 3.1
view	Login	RFOB 1 RFOB 4.1.1 RFOB 5 RFOB 5.1 RQOP 3.1
view	Lanciatore	RFOB 1 RFOB 7 RFOB 15 RFOB 16 RQOP 3.1
logic	Timer	RFOB 6
logic	ClientBackEnd	RFOB 3
logic	ClientWeb	RFOB 4.1.1 RFOB 7 RFOB 15
condivisi	DatiLogin	RFOB 1 RFOB 1.1 RFOB 5 RFOB 5.1
condivisi	Log	RFOB 1 RFOB 1.1 RFOB 3

10.2 Web

Package	Classe	Requisiti
view	DipendenteImpl	RFOB 4.1.1 RFOB 5 RFOB 5.1 RFOB 6 RFOB 6.1 RFOB 7

		RFOB 9 RFOB 10 RFOB 12 RFOB 13 RFOB 14 RFOB 15 RFOB 16
view	AmministratoreAziendaImpl	RFOB 4 RFOB 4.1 RFOB 4.1.1 RFOB 17 RFOB 18 RFOB 19 RFOB 20 RFOB 21 RFOB 22 RFOB 22.1 RFOB 23 RFOB 24 RFOB 25
view	AmministratoreSicurezzaImpl	RFOB 4 RFOB 4.1.1 RFOB 26 RFOB 27 RFOB 28 RFOB 28.1 RFOB 28.2 RFOP 29 RFOP 29.1 RFOB 30
control	ControlDipendente	RFOB 4.1.1 RFOB 5 RFOB 5.1 RFOB 6
control	ControlAmministratoreAzienda	RFOB 4.1.1
control	ControlAmministratoreSicurezza	RFOB 4.1.1
model	ClientBackEnd	RFOB 3
model	ModelTrofeo	RFOB 15.1
model	ModelDomanda	RFOB 27.1
model	ModelLogin	RFOB 5 RFOB 17 RFOB 26
model	ModelDipendente	RFOB 5 RFOB 5.1
model	ModelPunteggio	RFOB 15.1

model	ModelLog	RFOB 3
model	ModelRecupero	RFOB 4
condivisi	Domanda	RFOB 27.1
condivisi	Login	RFOB 5 RFOB 5.1
condivisi	Log	RFOB 3
condivisi	Trofeo	RFOB 15.1 RFOB 23.1 RFOB 24
condivisi	Dipendente	RFOB 12.1 RFOB 19.1 RFOB 21.1 RFOB 22.1
condivisi	Punteggio	RFOB 15.1 RFOB 23.1
condivisi	Recupero	RFOB 4

10.3 Mobile

Package	Classe	Requisiti
view	LoginActivity	
view	RecuperoPasswordActivity	
view	DashboardActivity	
view	TimerNotifica	
view	PunteggiActivity	
view	DomandaActivity	
view	DatipersonaliActivity	
controller	ControllerLogin	
controller	controllerPunteggi	
controller	ControllerDati	
controller	ControllerDomande	
controller	ControllerRecupero	
model	ClientBackEnd	RFOB 3
model	ModelDomanda	
model	ModelLogin	
model	ModelDipendente	
model	ModelPunteggi	
model	ModelLog	
model	ModelRecupero	

condivisi	Domanda	
condivisi	Login	
condivisi	Log	
condivisi	Dipendente	
condivisi	Punteggi	
condivisi	Recupero	

10.4 Back-end

Package	Classe	Requisiti
controller	ControllerLogin	
controller	ControllerDomanda	
controller	ControllerCalderone	
controller	ControlDipendenti	
controller	ControllerTrofei	
controller	ControllerDati	
controller	ControllerRecupero	
controller	ControllerPunteggi	
controller	ControllerLog	
checker	CheckLogin	
checker	CheckRispostaDomanda	
checker	CheckModificaDomanda	
checker	CheckDipendenti	
checker	CheckDati	
checker	CheckLog	
checker	CheckRecupero	
checker	CheckTrofei	
checker	CheckPunteggi	
access	AccessDipendenti	
access	AccessDomande	
access	AccessCalderone	
access	AccessPunteggi	
access	AccessLog	
access	AccessTrofei	
condivisi	Domanda	
condivisi	Login	
condivisi	Log	
condivisi	Trofeo	
condivisi	Dipendente	
condivisi	Punteggio	
condivisi	Recupero	

11 Tracciamento requisiti - componenti

Nelle tabelle se una o più celle riguardanti i componenti del sistema sono vuote, significa che quel componente non è delegato a gestire quel requisito.

11.1 Ambito Dipendente

Codice	Descrizione	Desktop	Web	Mobile	Back-end
RFOB 1	Il sistema dovrà lavorare in rete	view logic condivisi	view control model condivisi	view controller model condivisi	controller checker access condivisi
RFD 2	Il sistema assegnerà in automatico badge/trofei al Dipendente, quando questo raggiunge determinati obiettivi di Gamification				controller.ControllerTrofei checker.CheckTrofei access.AccessTrofei condivisi.Trofei
RFOB 3	Tutte le azioni di tutti gli utilizzatori, nel tempo, dovranno essere registrate nel sistema	condivisi.Log logic.ClientBackEnd model.ClientBackEnd	condivisi.Log model.ModelLog control.ControlBackEnd	condivisi.Log model.ModelLog controller.ControllerBackEnd	condivisi.Log controller.ControllerLog checker.CheckLog access.AccessLog
RFOB 4	Un Utente, che dovesse aver perso la propria password per accedere al sistema, potrà modificarla		model.ModelRecupero condivisi.Recupero	model.ModelRecupero controller.ControllerRecupero model.ModelRecupero condivisi.Recupero	controller.ControllerRecupero
RFOB 4.1	Un Utente, che dovesse aver eseguito correttamente la procedura di modifica password dimenticata, riceverà nella propria casella di posta elettronica una nuova password generata casualmente dal sistema			controller.ControllerRecupero	
RFOB 4.1.1	Al primo login la password dovrà essere modificata	view.Login logic.ClientBackEnd	view.DipendenteLogin control.ControlDipendente control.ControlAmministratoreAzienda	view.LoginActivity view.AmministratoreRecuperoPasswordActivity controller.ControllerRecupero	controller.ControllerRecupero

			control.ControlAmministratoreSicurezza		
RFOB 5	Un Dipendente dovrà potersi autenticare nel sistema inserendo le proprie credenziali di accesso	view.Login	view.DipendenteLoginActivity	view.LoginActivity	control.ControllerLogin
		condivisi.DatiLogin	control.ControlDipendente	control.ControllerLogin	
			model.ModelDipendente	model.ModelUserAccessDipendenti	
			condivisi.Login	condivisi.Login	
			model.ModelLogin		
RFOB 5.1	Un Dipendente effetuerà il login tramite web o tramite applicazione installata	view.Login	view.DipendenteLoginActivity	view.LoginActivity	
		condivisi.DatiLogin	control.ControlDipendente	control.ControllerLogin	
			model.ModelDipendente	model.ModelLogin	
			condivisi.Login	condivisi.Login	
RFOB 6	Il sistema dovrà sottoporre periodicamente ai Dipendenti Autenticati delle domande precaricate	view.Notifica	view.DipendenteTimerNotifica	view.TimerNotifica	
		logic.Timer	control.ControlDipendente		
RFOB 6.1	Una proposta di domanda dovrà essere completamente non invadente	view.Notifica	view.DipendenteTimerNotifica	view.TimerNotifica	
RFOB 7	Un Dipendente Autenticato potrà richiedere al sistema di venire sottoposto ad una domanda	view.Lancia	view.DipendenteDomanda	view.DomandaActivity	
		logic.ClientWeb			
RFOB 7.1	Un Dipendente Autenticato potrà richiedere un numero limitato di domande secondo il tempo trascorso				checker.CheckRispostaDoman
RFOB 8	Le domande che verranno sottoposte ai Dipendenti Autenticati dovranno essere di tipologia diversa				
RFOB 9	Le domande che verranno sottoposte ai Dipendenti Autenticati dovranno poter avere risposta direttamente dalla propria postazione di lavoro	view.ClientWeb	view.DipendenteDomanda	view.DomandaActivity	
RFOB 9.1	Le domande a cui verranno sottoposti i Dipendenti Autenticati forniti di dispositivi mobili aziendali potranno prevedere anche prove pratiche				

RFOB 9.2	La natura delle domande a cui verranno sottoposti i Dipendenti Autenticati potrà variare a seconda dell'impiego dello stesso				
RFOB 9.2.1	Il tipo delle risposte può essere: si/no, risposta chiusa, risposta multipla				
RFD 9.2.2	Il tipo delle risposte può essere: domanda con video, domanda con foto (entrambe con le modalità sopra citate), scansione di codice QR nel tempo limite, scansione di codici QR nell'ordine corretto				
RFOP 9.2.3	Il tipo delle risposte può essere: ordinamento di frasi o foto, trova errori nella foto, trova differenze nella foto, registra i codici RF-ID nell'ordine corretto				
RFOB 10	Un Dipendente Autenticato potrà modificare i propri dati personali, ovvero password e indirizzo di posta elettronica		view.DipendenteDatiPersonaliActivity		
RFOB 11	Quando un Utente Autenticato modifica i propri dati personali, il sistema si occuperà di inviargli una mail di conferma				controller.ControllerDati
RFOB 11.1	Quando un Utente Autenticato modifica la password di accesso alla sua area personale, il sistema gli invierà una mail di conferma				controller.ControllerDati
RFOB 11.2	Quando un Dipendente Autenticato modifica il suo indirizzo di posta elettronica, il sistema gli invierà una mail di conferma al nuovo indirizzo				controller.ControllerDati
RFOB 12	Un Dipendente Autenticato potrà visualizzare i propri dati personali memorizzati nel sistema		view.DipendenteDatiPersonaliActivity		
RFOB 12.1	I dati personali visualizzabili sono: nome, cognome, codice fiscale, indirizzo di posta elettronica, impiego, password, username		condivisi.Dipendente		access.AccessDipendenti

RFOB 13	Un Dipendente Autenticato a cui viene proposta la possibilità di rispondere ad una domanda potrà scegliere di visualizzarla e quindi rispondere, oppure rimandarla e rispondere successivamente	view.notifica	view.Dipendente	view.TimerNotifica	
RFOB 14	Una volta che una domanda viene mostrata ad un Dipendente Autenticato, questo dovrà necessariamente fornirle risposta		view.Dipendente	view.DomandaActivity	
RFOB 14.1	Il sistema sarà in grado di riconoscere se la domanda a cui verrà sottoposto un Dipendente Autenticato è una nuova domanda, una domanda posticipata o una domanda sbagliata in precedenza				checker.CheckRispostaDoman access.AccessDomande
RFOB 15	Un Dipendente Autenticato dovrà avere la possibilità di consultare le informazioni riguardanti i suoi punteggi	view.Lancia	view.Dipendente logic.ClientVedi	view.PuntiActivity condivisi.Punteggi	
RFOB 15.1	Le informazioni che un Dipendente Autenticato potrà consultare sono: storico delle risposte (sia corrette che errate), classifica aziendale, trofei/badge raccolti, statistiche derivanti dalle risposte		condivisi.Punteggio condivisi.TrofeoPunteggio model.ModelTrofeo model.ModelPunteggio	condivisi.PunteggiAccessPunteggi condivisi.Trofeo access.AccessTrofei	
RFOB 16	Un Dipendente Autenticato dovrà avere la possibilità di chiudere la propria sessione, senza terminare l'applicazione	view.Lancia	view.Dipendente	view.DashboardActivity	
RPOB 1	Il prodotto non dovrà risentire del traffico presente nella rete				
RPOB 2	Il prodotto dovrà essere prestazionale indipendentemente dalla posizione del server generale delle domande				

RQOB 2	Sarà fornita assieme al software documentazione adeguata riguardo l'installazione e l'utilizzo del sistema				
RQOB 2.1	Sarà fornito un manuale ad uso dei Dipendenti che coprirà gli aspetti di utilizzo del software dal loro punto di vista				
RQOB 3	Le interfacce che saranno visibili ai Dipendenti dovranno essere accattivanti e le funzionalità quanto più intuitive, così da invogliare l'utilizzo				
RQOP 3.1	Ogni elemento di interazione fornirà all'utilizzatore informazioni visive sulla possibile azione effettuabile	view	view	view	
RQOP 3.2	Da ogni schermata sarà possibile raggiungere un'area di aiuto che guiderà l'utilizzatore spiegandogli le operazioni effettuabili nella pagina in cui esso si trova		view	view	
RQOB 4	Tutti i processi di sviluppo aderiranno allo standard ISO/IEC 15504:1998 SPICE				
RQOB 5	Sarà fornita documentazione esaustiva di ogni metodo sviluppato				
RQOB 6	Le applicazioni web aderiranno allo stile W3C				
RQOB 6.1	Le applicazioni web aderiranno allo standard CSS3 definito da W3C				
RQOB 6.2	Le applicazioni web aderiranno allo standard XHTML1.1 definito da W3C				

11.2 Ambito Amministratore Azienda

Codice	Descrizione	Desktop	Web	Mobile	Back-end
RFOB 1	Il sistema dovrà lavorare in rete		view control		controller checker

			model condivisi		access condivisi
RFOB 1.1	Ogni azienda potrà accedere al sistema mediante connessione internet		view control model condivisi		controller checker access condivisi
RFOB 3	Tutte le azioni di tutti gli utilizzatori, nel tempo, dovranno essere registrate nel sistema		condivisi.Log model.ModelLog		checker.CheckLog controller.ControllerLog access.AccessLog condivisi.Log
RFOB 4	Un Utente, che dovesse aver perso la propria password per accedere al sistema, potrà modificarla		view.AmministratoreAziendaImpl		
RFOB 4.1	Un Utente, che dovesse aver eseguito correttamente la procedura di modifica password dimenticata, riceverà nella propria casella di posta elettronica una nuova password generata casualmente dal sistema				controller.ControllerRecuper
RFOB 4.1.1	Al primo login la password dovrà essere modificata		view.AmministratoreAziendaImpl		
RFOB 11	Quando un Utente Autenticato modifica i propri dati personali, il sistema si occuperà di inviargli una mail di conferma				controller.ControllerDati
RFOB 11.1	Quando un Utente Autenticato modifica la password di accesso alla sua area personale, il sistema gli invierà una mail di conferma				controller.ControllerDati
RFOB 17	Un Amministratore Azienda dovrà potersi autenticare nel sistema inserendo le proprie credenziali	model.ModelLogin	view.AmministratoreAziendaImpl		
RFOB 18	L'Amministratore Azienda Autenticato dovrà poter gestire gli account dei Dipendenti		view.AmministratoreAziendaImpl		

RFOB 19	L'Amministratore Azienda Autenticato dovrà poter registrare i Dipendenti inserendone i dati personali		view.AmministratoreAziendaImpl		
RFOB 19.1	I dati personali sono: nome, cognome, codice fiscale, email, impiego		condivisi.Dipendente		
RFOB 19.2	Ad ogni account Dipendente che verrà creato, il sistema dovrà inviargli i dati del suo account all'indirizzo email dichiarato utili per connettersi al sistema				controller.ControllerDipende
RFOB 19.2.1	Il sistema invia all'indirizzo email del nuovo Dipendente la sua username				controller.ControllerDipende
RFOB 19.2.2	Il sistema invia all'indirizzo email del nuovo Dipendente la sua password				controller.ControllerDipende
RFOB 20	L'Amministratore Azienda Autenticato potrà eliminare un account Dipendente		view.AmministratoreAziendaImpl		
RFOB 21	Un Amministratore Azienda Autenticato avrà la possibiltà di modificare i dati personali dei Dipendenti		view.AmministratoreAziendaImpl		
RFOB 21.1	I campi che potranno essere modificati sono: nome, cognome, indirizzo email, impiego, codice fiscale		condivisi.Dipendente		access.AccessDipendente
RFOB 22	L'Amministratore Azienda Autenticato dovrà poter visualizzare i dati di ogni singolo Dipendente		view.AmministratoreAziendaImpl		
RFOB 22.1	I dati visualizzabili sono: nome, cognome, codice fiscale, indirizzo email, impiego, nickname		condivisi.Dipendente view.AmministratoreAziendaImpl		
RFOB 23	L'Amministratore Azienda Autenticato dovrà poter visionare le statistiche personali di ogni singolo Dipendente		view.AmministratoreAziendaImpl		
RFOB 23.1	Le statistiche personali sono: punti, trofei/badge		condivisi.Punteggio condivisi.Trofeo		

RFOB 24	L'Amministratore Azienda Autenticato dovrà poter modificare i badge/trofei di ogni singolo Dipendente		condivisi.Trofeo view.AmministratoreAziendaImpl		
RFOB 25	Un Amministratore Azienda Autenticato dovrà avere la possibilità di terminare la propria sessione		view.AmministratoreAziendaImpl		
RPOB 1	Il prodotto non dovrà risentire del traffico presente nella rete				
RQOB 2	Sarà fornita assieme al software documentazione adeguata riguardo l'installazione e l'utilizzo del sistema				
RQOB 2.2	Sarà fornito un manuale ad uso Amministratore Azienda riguardo tutte le azioni che potrà compiere				
RVOB 1	Il sistema dovrà essere funzionale presso aziende diverse, anche con caratteristiche molto differenti fra di loro				

11.3 Ambito Amministratore Sicurezza

Codice	Descrizione	Desktop	Web	Mobile	Back-end
RFOB 1	Il sistema dovrà lavorare in rete	view logic condivisi	view control model condivisi	view controller model condivisi	controller checker access condivisi
RFOB 3	Tutte le azioni di tutti gli utilizzatori, nel tempo, dovranno essere registrate nel sistema		condivisi.Log model.ModelLog		checker.CheckLog controller.ControllerLog access.AccessLog condivisi.Log
RFOB 4	Un Utente, che dovesse aver perso la propria password per accedere al sistema, potrà modificarla		view.AmministratoreSicurezzaImpl		

RFOB 4.1	Un Utente, che dovesse aver eseguito correttamente la procedura di modifica password dimenticata, riceverà nella propria casella di posta elettronica una nuova password generata casualmente dal sistema				controller.ControllerRecupero
RFOB 4.1.1	Al primo login la password dovrà essere modificata		view.AmministratoreSicurezzaImpl		
RFOB 11	Quando un Utente Autenticato modifica i propri dati personali, il sistema si occuperà di inviargli una mail di conferma				controller.ControllerDati
RFOB 11.1	Quando un Utente Autenticato modifica la password di accesso alla sua area personale, il sistema gli invierà una mail di conferma				controller.ControllerDati
RFOB 26	Un Amministratore Sicurezza dovrà potersi autenticare nel sistema inserendo le proprie credenziali	model.ModelLogin	view.AmministratoreSicurezzaImpl		
RFOB 27	Un Amministratore Sicurezza Autenticato avrà la possibilità di consultare le domande dell'azienda		view.AmministratoreSicurezzaImpl		
RFOB 27.1	La consultazione farà vedere: il tipo, il testo, le risposte possibili, le risposte corrette, le immagini / video (se presenti), la categoria a cui verrà somministrata, la piattaforma, la temporizzazione (se presente) e il punteggio della domanda	model.ModelDomanda	condivisi.Domanda	access.AccessDomanda access.AccessCalderone	
RFOB 28	Un Amministratore Sicurezza Autenticato dovrà poter modificare l'insieme delle domande di un'azienda		view.AmministratoreSicurezzaImpl		
RFOB 28.1	Un Amministratore Sicurezza Autenticato avrà la possibilità di aggiungere una domanda, dall'insieme di quelle già presenti nel database generale		view.AmministratoreSicurezzaImpl		

RFOB 28.2	Un Amministratore Sicurezza Autenticato avrà la possibilità di rimuovere una domanda dal database di domande da somministrare ai Dipendenti		view.AmministratoreSicurezzaImpl		
RFOP 29	L'Amministratore Sicurezza Autenticato dovrà poter modificare gli obiettivi di Gamification		view.AmministratoreSicurezzaImpl		
RFOP 29.1	L'Amministratore Sicurezza Autenticato dovrà poter modificare i badge o trofei assegnabili agli utenti al raggiungimento di obiettivi di Gamification		view.AmministratoreSicurezzaImpl		
RFOB 30	Un Amministratore Sicurezza Autenticato dovrà avere la possibilità di terminare la propria sessione		view.AmministratoreSicurezzaImpl		
RPOB 1	Il prodotto non dovrà risentire del traffico presente nella rete				
RQOB 2	Sarà fornita assieme al software documentazione adeguata riguardo l'installazione e l'utilizzo del sistema				
RQOB 2.4	Sarà fornito un manuale ad uso Amministratore Sicurezza riguardo le azioni che potrà compiere				

12 Prototipi di interfaccia utente

In questo capitolo saranno presentate alcune delle funzionalità del prodotto tramite prototipi di interfaccia. La realizzazione di questi prototipi ha seguito il più possibile i canoni di *interfaccia intuitiva*. Questo dovrebbe garantire una semplice comprensione delle funzionalità.

12.1 Interfaccia desktop

Questa parte dell'applicazione è riservata ai soli Dipendenti, i quali potranno autenticarsi tramite Login o accedere alla procedura di recupero password (Figura 12).

Figura 12: schermata di Login per l'applicazione desktop

Una volta effettuato il Login l'applicazione resterà in background per una certa quantità di tempo, al termine del quale apparirà nuovamente il pop-up che inviterà a rispondere ad una nuova domanda (Figura 13).

Figura 13: schermata di Notifica di una nuova domanda

Scegliendo di rispondere subito, l'applicazione aprirà il browser consentendo di ripondere ad una nuova domanda (tramite l'applicativo web). Diversamente, rimandando la risposta la domanda verrà messa in attesa, per essere riproposta successivamente.

Sia in caso di risposta che in caso di posticipo, l'applicazone resterà in background fino alla successiva scadenza del timer.

12.2 Interfaccia web

A differenza dell'applicazione desktop, l'applicazione web sarà accessibile sia ai Dipendenti sia agli Amministratori. Per questo motivo il Login (Figura 14) dovrà rendere disponibili diversi servizi a seconda della tipologia di Utente collegato.

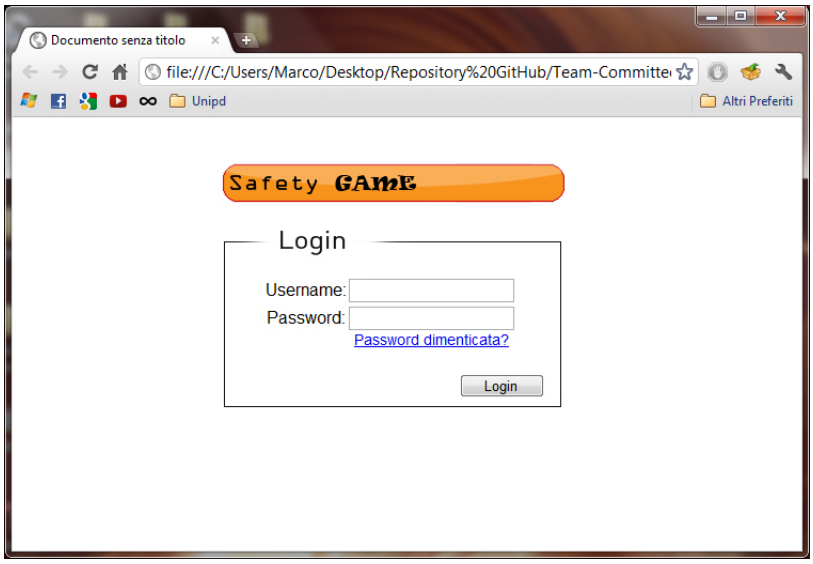

Figura 14: schermata di Login per l'applicativo web

Nel caso l'Utente sia un Dipendente, l'applicativo web permetterà di accedere ai relativi servizi, i medesimi disponibili nell'applicazione mobile.

Nel caso l'Utente sia un Amministratore, questi potrà accedere alle funzionalità di gestione.

In Figura 15 possiamo vedere come un Amministratore Azienda possa eliminare o aggiungere un Utente oppure modificarne i dati o visualizzarne le statistiche.

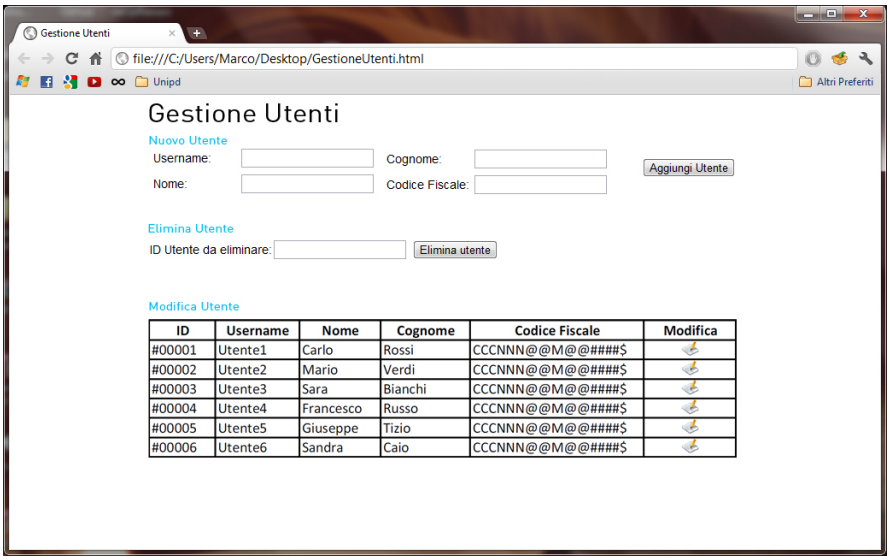

Figura 15: schermata per la gestione degli utenti

Inoltre, l'Amministratore Sicurezza potrà scegliere nuove domande da poter proporre, oppure rimuoverne altre già disponibili. Per fare questo vi saranno due schermate differenti.

La prima (Figura 16) permetterà di scegliere quali domande rendere disponibili tramite checkbox, per poi dare conferma tramite Button.

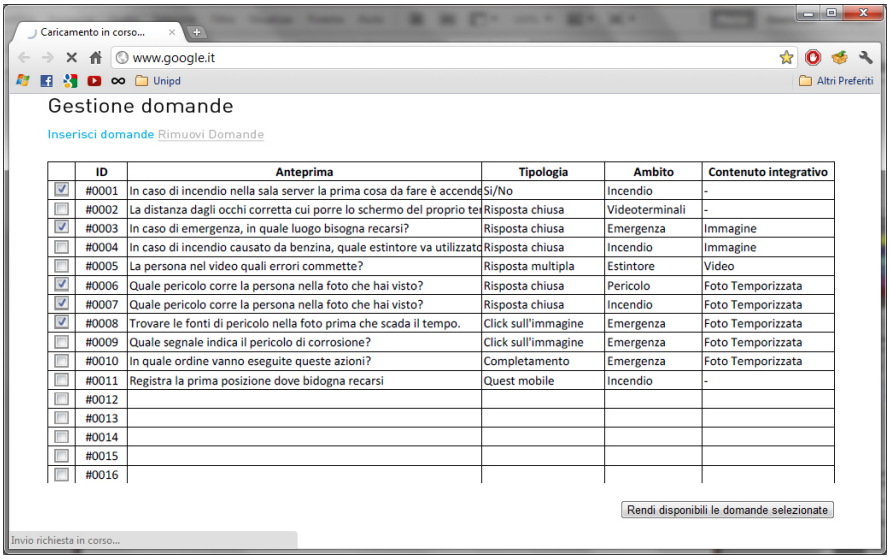

Figura 16: schermata per l'inserimento di nuove domande

La seconda schermata (Figura 17) permetterà di scegliere quali domande, fra quelle già disponibili, rimuovere dal proprio set, sempre selezionando tramite Checkbox e dando conferma con Button.

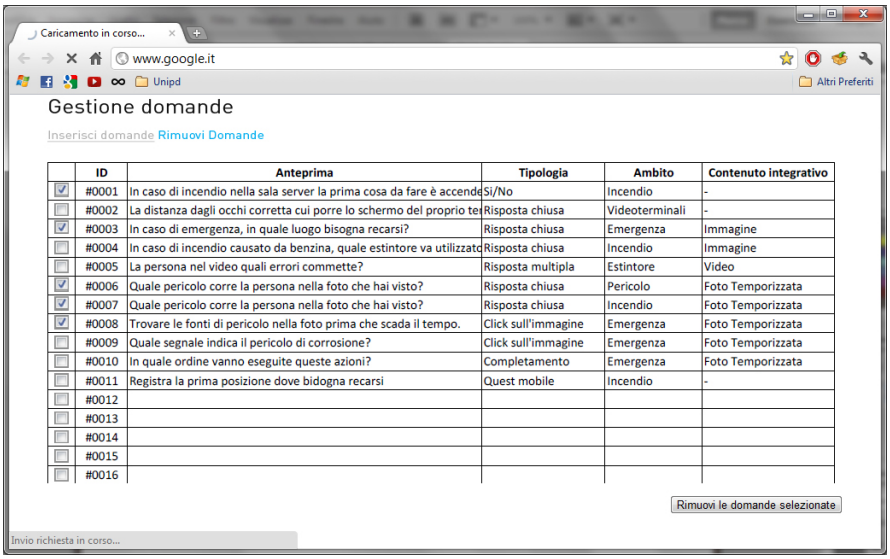

Figura 17: schermata per la rimozione di domande già presenti

12.3 Interfaccia Mobile

L'interfaccia dell'applicazione mobile sarà strutturata indicativamente come segue.

- **Login e Notifica** (Figura 18):

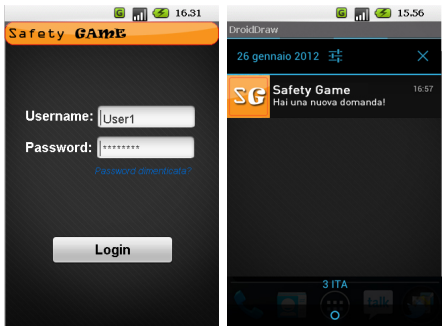

Figura 18: Schermate di Login e di Notifica per l'applicazione mobile

La schermata di Login permette ai Dipendenti di effettuara l'accesso. Un Amministratore non potrà effettuare l'accesso tramite l'applicazione mobile.

Come per gli altri Login, sarà possibile avviare la procedura per il recupero password.

Per le notifiche si utilizzerà il sisitema di notifiche previsto dall'ambiente Android.

- **Rispondere ad una domanda** (Figura 19):

Figura 19: Risposta ad una domanda

La schermata per fornire una risposta non sarà standard, ma varierà a seconda della tipologia di domanda. Ad ogni modo, saranno disponibili degli elementi di scelta ed un bottone per l'invio della risposta.

- Visualizzazione dati Utente e modifica password (Figura 18):

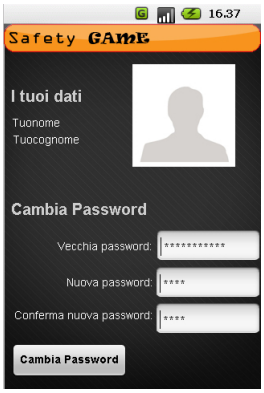

Figura 20: schermata per visualizzare i propri dati e per cambiare la password.

Ogni utente avrà la possibilità di visualizzare i propri dati e di modificare la password tramite apposito form. Il tasto per confermare il cambio password sarà posizionato in modo da ridurre al minimo il rischio di invio involontario.

- Visualizzazione Punteggi e Badge (Figura 18):

Figura 21: schermata per la visualizzazione di punteggi e Badge

In questa schermata l'Utente avrà la possibilità di visualizzare i propri punteggi e i premi (o Badge) ricevuti.

13 Stime di fattibilità e di bisogno di risorse

L'analisi accurata dell'architettura progettata ha permesso di constatare che le tecnologie che si è scelto d'adottare risultano essere sufficientemente adeguate per la realizzazione del prodotto e riescono a ricoprire le esigenze progettuali. Nonostante non tutti gli strumenti scelti siano conosciuti dalla totalità del gruppo, la scelta di utilizzare linguaggi simili tra loro come JSP - Java - Android SDK ha permesso di facilitare l'apprendimento e migliorare l'integrità dei vari componenti di cui è costituito il prodotto.
Il reperimento delle risorse da impiegare nella realizzazione non si è rivelato problematico o complesso in quanto si è scelto di impiegare strumenti completamente gratuiti. Gli strumenti impiegati sono, come già citato, JSP per la generazione delle pagine dinamiche dell'applicazione web, Java per la realizzazione dell'applicazione Desktop ed Android SDK per la realizzazione dell'applicazione mobile; o ancora l'IDE Eclipse o BOUML.