# T5 Backend:

| Nome | Tipo | Modifiche |
|---|---|---|
| **Game/GameController.java** | | |
| **GameFactoryFunction** | Interfaccia | Aggiunta dei parametri per la gestione della scalata. |
| **RegisterGames** | Metodo | Aggiunta della registrazione di scalata. |
| **StartGame** | Metodo | Aggiunti parametri opzionali per la scalata<br>Aggiunta gestione della modalità "Scalata" con creazione dell'oggetto ScalataGame e restituzione dati utili a UtilEditor.<br>Il controllo della partita esistente tiene conto se la scalata sta effettuando una transizione al round successivo. |
| **TotalScore** | Variabile | Fondamentale per calcolare il punteggio totale della scalata. |
| **processScalataParameters** | Metodo | Chiamata al metodo implementata ScalataGame. |
| **buildScalataResponse** | Metodo | Costruisce la risposta da inviare in caso di partita in modalità scalata. |
| **gestisciPartita** | Metodo | Migliorato controllo di fine partita, in modo da distinguere termine sfida e termine scalata. Aggiunta restituzione del punteggio totale della scalata. |
| **Game/GameLogic.java** | | |
| **PlayTurn** | Metodo Astratto | Aggiunto il booleano isRoundEnd per verificare la fine di un round. |
| **CreateGame** | Metodo | Aggiornato il vecchio metodo.<br>Aggiunto un CreateGame(int scalataID) per gestire la creazione della partita con un ID Scalata. |
| **GameLogic** | Classe | Aggiunta setter/getter. Ritorna il PlayerID, tipo di robot, difficoltà, classe. |
| **Game/ScalataGame.java** | | |
| **PlayTurn** | Metodo | Aggiunto boolean isRound End per controllare la fine del round.<br>Modificato per gestire la logica della scalata nei turni con l'aggiunta del controllo sullo stato della scalata ( IN_PROGRESS, WIN, LOST) e gestione dell'aggiornamento del turno. |
| **GetScore** | Metodo | Aggiunta dell'aggiornamento del totalScore. |
| **ScalataGame** | Classe | Aggiunta metodi getter/setter per id_scalata, currentRoundID, currentGameID, currentTurnID |

| | | |
|---|---|---|
| **ProcessScalataParameters** | Metodo | Aggiunta del metodo per “normalizzare” i parametri che sono passati come array json. |
| **CheckGame** | Metodo | Viene fatto il controllo mediante round corrente. |
| **nextRoundTransition** | Attributo/ Metodo | Permette di notificare all’esterno quando la scalata passa al round successivo. |
| **toggleRoundTransition** | Metodo | Imposta il booleano nextRoundTransition rendendolo true quando si passa al round successivo in playTurn di ScalataGame, e false quando viene fatto il controllo al round successivo in startGame nel GameController. |
| **isRoundTransition** | Metodo | Ritorna il valore di nextRoundTransition |
| **Game/Sfida.java** | | |
| **PlayTurn** | Metodo | Abbiamo effettuato la modifica perché è un metodo ereditato da GameLogic e dato che abbiamo il playTurn in GameLogic abbiamo adattato il metodo con l’aggiunta del booleano isRoundEnd.<br><br>Sono state aggiunte le chiamate a createTurn, EndRound ed endGame. |
| **GameCoverage** | Variabile | Aggiunta variabile per ottenere la copertura del codice tramite getScore. Aggiunto anche il relativo get. |
| **GetScore** | Metodo | Imposta la variabile gameCoverage con la copertura ricevuta in ingresso. |
| **Interfaces/BaseService.java** | | |
| **CallRestPut** | Metodo | Aggiunte per un put di un jsonObject. |
| **Interfaces/T4Service.java** | | |
| **registerAction("CreateGame")** | Registrazione azione REST | Aggiunta del parametro scalataId di tipo Optional<Integer> per la creazione di un gioco, modificando la chiamata REST per includerlo. |
| **registerAction("CreateRound")** | Registrazione azione REST | Aggiunto il parametro robot_id di tipo int per la creazione di un round, per includere l'ID del robot. |
| **registerAction("EndTurn")** | Registrazione azione REST | Modifica dei parametri (ordine cambiato da (String, String, int) a (int, String, String)) e aggiunta la gestione del robot ID. |
| **registerAction("GetRobotID")** | Registrazione azione REST | Aggiunta di una nuova azione REST per recuperare l'ID del robot. |

| | | |
|---|---|---|
| **CreateTurn** | Metodo privato REST (chiamata POST) | Aggiunta gestione del response JSON. |
| **EndTurn** | Metodo privato REST (chiamata PUT) | Conversione del payload da MultiValueMap<String, String> a JSONObject per inviare dati in formato JSON.<br>L'ID del turno (turnId) ora è una String invece di int. |
| **registerAction("UpdateScalata")** | Registrazione azione REST | Aggiunta la registrazione per l'aggiornamento della scalata. |
| **registerAction("CloseScalata")** | Registrazione azione REST | Aggiunta la registrazione per la chiusura di una scalata. |
| **CreateGame** | Metodo privato REST (chiamata POST) | Aggiunta della logica per inviare i dati di creazione del gioco, inclusa la gestione del parametro scalataId opzionale. |
| **EndGame** | Metodo privato REST (chiamata POST) | Metodo rimasto invariato ma invece di utilizzare MultiValueMap<String, String> formData abbiamo utilizzato un JSONObject obj. |
| **CreateRound** | Metodo privato REST (chiamata POST) | Aggiunta del parametro robot_id per la creazione di un round. |
| **EndTurn** | Metodo privato REST (chiamata PUT) | Modificato per inviare i dati di fine turno come JSON tramite una chiamata PUT. |
| **GetRobotID** | Metodo privato REST (chiamata GET) | Nuovo metodo per ottenere l'ID del robot, usando i parametri classUT, robot_type e difficulty in una chiamata GET a /robots. |
| **CreateScalata** | Metodo privato REST (chiamata POST) | Nuovo metodo per creare una scalata, che invia i dati come JSON e restituisce l'ID della scalata creata. |
| **UpdateScalata** | Metodo privato REST (chiamata PUT) | Aggiunta la logica per aggiornare una scalata, modificando il round corrente e la data di aggiornamento. |
| **CloseScalata** | Metodo privato REST (chiamata PUT) | Aggiunta la logica per chiudere una scalata, inviando il risultato finale, il punteggio e la data di chiusura tramite una chiamata PUT. |
| **Test/Interfaces/T4ServiceTest,java** | | |
| **testCloseScalata_validClose** | Test Aggiunto | Test utilizzato per verificare la corretta chiusura di una scalata |

| | | |
|---|---|---|
| **testUpdateScalata_validUpdate** | Test aggiunto | Test aggiunto per verificare il corretto aggiornamento di una scalata |
| **testCreateScalata_ValidResponse** | Test modificati | Corretto il nome della chiave JSON da “scalataId”a “id” e la rotta API dove effettuare la richiesta(verso /scalates)<br>Aggiunto controllo del JSON object inviato<br>Il confronto all’interno dell’assert verrà effettuato solo sull’ID e non sull’intero JSON |
| **testCreateGame_ValidGameCreation** | Test modificati | Aggiunto id della scalata come parametro opzionale di test, all’interno della chiamata a handleRequest |
| **testCreateRound_ValidResponse** | Test modificati | Aggiunto id del robot all’interno dell’handleRequest |
| **testGetRisultati_ValidParameters** | Test Modificati | Aggiunta verifica del JSON inviato |
| **testCreateTurn_ValidResponse** | Test Modificati | Corretto il nome della chiave JSON da “turnID” a “id”.<br>Viene confrontato solo l’id all’interno dell’assert |
| **testEndTurn_SuccessRespons** | Test Modificati | Aggiunta verifica del JSON inviato<br>turnID viene inviatgo come stringa |
| **testEndGame_ValidGameEnd** | Test Modifcati | Corretto metodo HTTP da POST a PUT<br>Aggiunta verifica del JSON inviato |
| **testEndGame_ValidGameEnd_NotWinner** | Test Modificati | Aggiunta verifica del JSON inviato |

## T5 Frontend:

| Nome | Tipo | Modifiche |
|---|---|---|
| **js/Button_Editor.js** | | |
| **DOMContentLoaded** | Event Listener | Aggiunto un controllo if per verificare se modalita è "Scalata". Se modalita è "Scalata", aggiunge round_corrente_info nel messaggio. |
| **flush_localStorage** | Funzione | Se modalita è "Scalata", vengono eliminati anche i dati specifici della scalata (scalataId, SelectedScalata, gameId, ecc.). |
| **js/REST_Editor.js** | | |
| **GetGameData** | Funzione | Introdotto controllo per modalita == "Scalata", Aggiunto is_scalata_inprogress per verificare se la scalata è in corso, Se la scalata non è iniziata, chiama handleScalataParameters(), aggiunti nuovi parametri nel return (scalata_name, scalata_classes, ecc.).<br>Riposizionato controllo sul parametro del link. |
| **$(document).ready** | Event Listener | Aggiunto if(is_scalata_inprogress == false) prima di chiamare startGame(data).<br>Aggiunto try-catch, dove in caso di eccezione viene mostrato un alert che porta al menu principale. |
| **handleResponse** | Funzione | Aggiunto totalScore tra i parametri estratti da response. |
| **processCoverage** | Funzione (async) | Aggiunto totalScore tra i parametri della funzione. |

| | | |
|---|---|---|
| **handleEndGame** | Funzione | Modifiche radicali per migliorare la gestione della modalità “Scalata”. Aggiunta condizione di vittoria di una partita scalata, e degli alert a seconda della vittoria, sconfitta, passaggio al round successivo. Ad ogni alert è seguito l’inserimento di un timeout di 10 secondi che reindirizza in una pagina di default. |
| **js/Util_Editor.js** | | |
| **startGame** | Funzione async | Se modalita === "Scalata", recupera roundID, gameID, turnID, scalataID dalla risposta JSON e li salva in localStorage. |
| **getScalataClasse** | Funzione | Estrae la classe associata al round corrente dalla JSON della scalata. |
| **getScalataRobot** | Funzione | Estrae il robot associato al round corrente dalla JSON della scalata. |
| **getScalataDifficulty** | Funzione | Estrae e converte la difficoltà del round corrente. |
| **GetDifficulty** | Funzione | Mappa la difficoltà da Beginner, Intermediate, Advanced a valori numerici 1, 2, 3. |
| **HandleScalataParameters** | Funzione | Recupera e stampa i dati relativi alla scalata da localStorage. |
| **HandleScalataNextRound** | Funzione | Controlla se la scalata è terminata, Se la scalata è ancora in corso, aggiorna localStorage con i parametri del prossimo round, e Incrementa current_round_scalata e aggiorna il punteggio della scalata per la visualizzazione nel Frontend. |
| **js/Var_Editor.js** | | |
| **var editor** | Variabili globali | Aggiunte le variabili per gestire la modalità Scalata (ID, nome, round corrente, totale round, robot, difficoltà, classi, punteggio totale, stato della scalata). |
| **js/gamemode_scalata.js** | | |
| **gamedatawriter** | AJAX e gestione scalata | Aggiunta chiamata AJAX per ottenere la lista delle scalate e pulsante "Info" per identificare se la scalata è predefinita o personalizzabile. |
| **PressedSubmit** | Funzione | Rimossa la chiamata AJAX per salvare la scalata e aggiunto il reindirizzamento all'editor basato su ClassUT in localStorage. |
| **RetrieveScalata** | Funzione | Aggiunta per recuperare i dati di una scalata, salvandoli in localStorage (classi, robot, difficoltà, numero di round, ecc.). |

# T4:

| Nome | Tipo | Modifiche |
|---|---|---|
| **round/round.go** | | |
| **Round** | Struct | Aggiunti RobotID e Robot alla struttura. |
| **CreateRequest** | Struct | Aggiunta proprietà RobotID con validazione per obbligatorietà. |
| **CreateRequest** | Funzione | Controllo sul id del robot. |
| **UpdateRequest** | Struct | Aggiunto id del robot. |
| **FromModel** | Funzione | Modificata per includere RobotID dalla struttura del modello. |
| **round/Service.go** | | |
| **Create** | Funzione | Ora assegna un solo robot al round, aggiungendo il campo RobotID e utilizzando l'ID del robot nel momento della creazione del round. |
| **Update** | Funzione | Aggiunta la modifica del metodo, passando da Updates(req) a Updates(r) e modificato il trattamento dell'errore. |
| **FindById** | Funzione | Aggiunta la pre-caricamento del campo Robot. |
| **FindByGame** | Funzione | Aggiunta la pre-caricamento del campo Robot. |

# T1 Frontend:

| Nome | Tipo | Modifiche |
|---|---|---|
| **js/Scalata.js** | | |
| **handleRoundsChange, handleConfirm** | Funzioni | Modifica effettuata per impedire l'inserimento del numero del round da tastiera.<br>Inoltre tutti i controlli sul nome della scalata e sul numero di round sono stati spostati sul tasto aggiungi in handleRoundsChange(), di conseguenza sono stati tolti da handleConfirm() che posizionava tali controlli solo alla fine della creazione della scalata (click tasto conferma). |
| **SelectClasses** | Funzione | Aggiunti gestori di eventi per i menu a tendina delle card con i relativi controlli. |
| **DisplayClasses** | Funzione | Modifiche effettuate per mostrare le coperture da battere, robot e difficoltà. |
| **submitScalataData** | Funzione | Aggiunta la riabilitazione del tasto "aggiungi" e dei componenti della parte superiore della pagina come il nome, descrizione e numero di round. |
| **ready** | Funzione | Aggiunto il codice per modificare il cambiamento di modalità scalata, nel caso di scalata predefinita setta il robot selezionato dall'amministratore. |
| **updateDifficultySelect** | Funzione | Aggiunta per cambiare le difficoltà in base al robot selezionato, perché nel caso un robot abbia una difficoltà con coverage nulla non viene proprio mostrata all'amministratore. |
| **$(document).on('change', '.robot-select')** | Event Listener | Quando cambia il robot selezionato viene chiamata updateDifficultySelect. |
| **js/class.js** | | |
| **EliminaScalata** | Funzione | Nuova funzione per eliminare una scalata tramite la chiamata DELETE all' all'endpoint /delete_scalata/{scalataName}. |
| **EliminaClasse** | Funzione | Aggiunto controllo sullo **status 409** (classe usata in una scalata).<br>Analizza il testo della risposta per estrarre i nomi delle scalate associate.<br>Utilizza **SweetAlert (swal)** per mostrare un messaggio di conferma all'utente.<br>Se l'utente conferma, elimina prima le scalate associate e poi la classe. |
| **js/scalata_tour.js** | | |
| **#modeRobotSelect** | Elemento UI | Aggiunto nuovo step nel tour per spiegare la selezione tra scalata personalizzabile e predefinita. |
| **localStorage.getItem('tourShown')** | Controllo | Ora il tour viene mostrato solo la prima volta grazie all'uso di localStorage. |
| **#summaryButton** | ID | Corretto da #summarymButton a #summaryButton. |

# T1 Backend:

| Nome | Tipo | Modifiche |
|---|---|---|
| **controller/HomeController.java** | | |
| **EliminaClasse** | Metodo | Aggiunto il controllo per verificare se la classe è presenta in una scalata prima di eleminarla. |
| **service/AdminService.java** | | |
| **UploadTest** | Metodo | Ora passa l'oggetto ClassUT a RobotUtil.saveRobots invece del solo nome della classe.<br>Aggiunta lista robotList con i robot disponibili (Randoop, Evosuite) e associata all'oggetto ClassUT.<br>Aggiunta lista robotDifficultyList con i livelli di difficoltà (Beginner, Intermediate, Advanced) e associata all'oggetto ClassUT. |
| **modificaClasse** | Metodo | Aggiunti i campi robot, robotDifficulty e coverage all'Update della classe.<br>Ora aggiorna anche le liste di robot e difficoltà oltre alle informazioni di base. |
| **service/ScalataService.java** | | |
| **FindScalataByClassName** | Metodo | Aggiunto metodo per ritornare la lista delle scalate che contengono una data classe. |
| **repository/ScalataRepository.java** | | |
| **FindBySelectedClassesClassName** | Metodo | Aggiunto nuovo metodo per trovare tutte le scalate che contengono una classe specifica. |
| **model/ClassUT.java** | | |
| **RobotList** | Variabile | Aggiunta lista per memorizzare i tipi di robot associati alla classe. |
| **RobotDifficulty** | Variabile | Aggiunta lista per memorizzare i livelli di difficoltà dei robot. |
| **Coverage** | Variabile | Aggiunta lista per memorizzare la copertura dei test della classe. |
| **ClassUT** | Costruttore | Modificato per accettare i nuovi parametri robotList, robotDifficulty, e coverage. |
| **getRobotList, setRobotList** | Getter/ Setter | Permette di ottenere e impostare la lista dei robot associati alla classe. |
| **getRobotDifficulty, setRobotDifficulty** | Getter/ Setter | Permette di ottenere e impostare la lista dei livelli di difficoltà dei robot. |
| **getCoverage, setCoverage** | Getter/ Setter | Permette di ottenere e impostare la lista delle coperture di test della classe. |
| **ToString** | Metodo | Modificato per includere robotList, robotDifficulty e coverage nella stringa di output. |

| filesystem/RobotUtil.java | | |
|---|---|---|
| **SaveRobots** | Metodo | Ora riceve un oggetto ClassUT invece del solo className, permettendo di aggiornare direttamente i dati della classe.<br>Aggiunta gestione delle coperture (coverageList[]) per memorizzare i risultati dei test in MongoDB.<br>Ora imposta le coperture dei test (setCoverage(...)) direttamente su classUT. |
| **Model/SelectedClasses.java** | | |
| **SelectedClasses** | Classe | Nuova classe per rappresentare una classe selezionata in una scalata, con informazioni su robot e difficoltà. |
| **ClassName** | Variabile | Nome della classe selezionata. |
| **robot** | Variabile | Tipo di robot associato alla classe. |
| **difficulty** | Variabile | Livello di difficoltà associato alla classe. |
| **SelectedClasses()** | Costruttore | Inizializza un'istanza con nome classe, robot e difficoltà. |
| **getClassName, setClassName** | Getter/Setter | Permette di ottenere e impostare il nome della classe. |
| **getRobot, setRobot** | Getter/Setter | Permette di ottenere e impostare il robot associato. |
| **getDifficulty, setDifficulty** | Getter/Setter | Permette di ottenere e impostare la difficoltà associata. |