

Università degli Studi di Salerno

Corso di Ingegneria del Software

AstroMark

Test Regression Report

Versione 1.0



Data: 30/01/2025

Progetto: AstroMark	Versione: 1.0
Documento: Test Regression Report	Data: 30/01/2025

Partecipanti:

Nome	Matricola
Giuseppe Cavallaro	0512116926
Mario Cosenza	0512116320
Mario Fasolino	0512116965
Giulio Sacrestano	0512116812

Scritto da:	Mario Cosenza

Revision History

Data	Versione	Descrizione	Autore
30/01/2025	1.0	Stesura finale del documento	Mario Cosenza

Indice

1. Introduzione 4

2. Relazione con altri documenti 4

3. Test log..... 5

3.1 TC4..... 5

4. Glossario..... 8

1. Introduzione

Il report di test di regressione, presentato di seguito, fornisce una valutazione della qualità dell'applicazione AstroMark a seguito delle recenti modifiche al codice. Grazie all'integrazione continua, siamo in grado di garantire che ogni nuova versione del software sia sottoposta a rigorosi controlli di qualità. I test di regressione, eseguiti automaticamente dalla nostra pipeline CI, hanno verificato che le nuove funzionalità introdotte non abbiano introdotto regressioni nelle funzionalità esistenti.

2. Relazione con altri documenti

Il presente progetto si basa sull'analisi e il confronto con piattaforme di gestione della didattica, già consolidate e affermate nel settore, le quali hanno dimostrato notevole efficacia. Tra queste, un punto di riferimento significativo è rappresentato dalle soluzioni sviluppate da Argo per la gestione della didattica.

Di seguito si presenta un elenco dei documenti chiave del progetto a cui si fa esplicito riferimento:

- **Test Plan (TP):** documento di pianificazione della fase di test del sistema.
- **Test Case Specification (TCS):** documento che descrive in dettaglio una serie di test case, specificando input, azioni, condizioni di esecuzione e risultati attesi.
- **Test Execution Report:** documento di log dell'esecuzione dei test.

Oltre ai documenti del progetto, si fa riferimento ad opere di letteratura tecnica che hanno contribuito allo sviluppo metodologico e concettuale di questo lavoro:

- **Object-Oriented Software Engineering Using UML, Patterns, and Java™ Third Edition** di Bernd Bruegge & Allen H. Dutoit.

3. Test log

In questa sezione è mostrato il risultato del test di regressione eseguito su TC4. L'intera suite è stata nuovamente testata ma per brevità sono omessi i risultati degli altri TC, essendo tutti "Passati", è possibile comunque reperire i log delle esecuzioni dal repository di AstroMark.

3.1 TC4

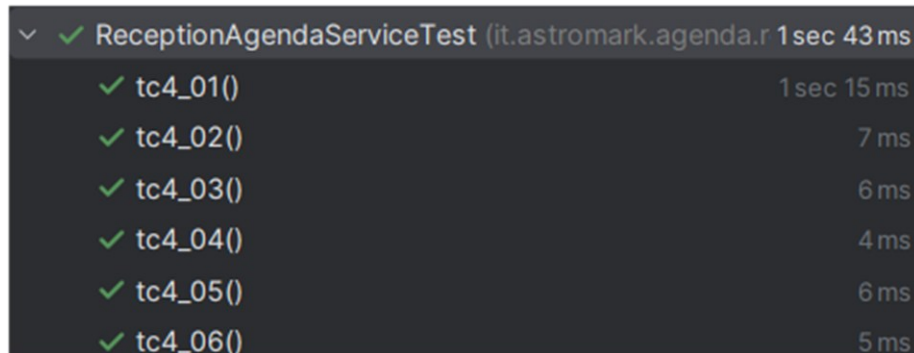
A seguito del fallimento di TC4 è iniziata una fase di debugging. Analizzando il metodo "book", definito in ReceptionAgendaServiceImpl, risultava mancante nel builder l'inizializzazione dell'id del booking, non essendo questo autogenerato, ma dato dalla combinazione dell'id del genitore e dello slot che si intende prenotare. Questa falla non è stata evidenziata nel test di unità, poiché eseguito in un contesto dove la validazione tramite il modulo Jakarta, preposto per le entità, non è attiva.

Metodo corretto

```
@Override 13 usages  Mario Cosenza *
@Transactional
@PreAuthorize("hasRole('PARENT')")
public boolean book(Integer receptionTimeslotID) {
    var slot : ReceptionTimeslot = receptionTimeslotRepository.findByIdAndDateAfter(receptionTimeslotID, LocalDate.now());
    var parent : Parent = authenticationService.getParent().orElseThrow();
    for (var student : parent.getStudents()) {
        if (slot.getReceptionTimeTable().getTeacher().getTeacherClasses().stream().anyMatch( TeacherClass c -> c.getSchoolClass().getStudents().contains(student))) {
            if (slot.getBooked() < slot.getCapacity()) {
                slot.setBooked((short) (slot.getBooked() + 1));
                receptionBookingRepository.save(ReceptionBooking.builder()
                    .bookingOrder(slot.getBooked())
                    .id(new ReceptionBookingId(parent.getId(), receptionTimeslotID))
                    .parent(parent)
                    .confirmed(false)
                    .refused(false)
                    .receptionTimeslot(slot).build());
                receptionTimeslotRepository.save(slot);
                return true;
            }
        }
    }
    return false;
}
```

Unit test

I test unitari condotti sul servizio di prenotazione del ricevimento, in particolare sul metodo "book" della classe ReceptionAgendaServiceImpl, hanno prodotto esiti positivi essendo il risultato conforme con quello atteso. I test sono dunque "Passati".

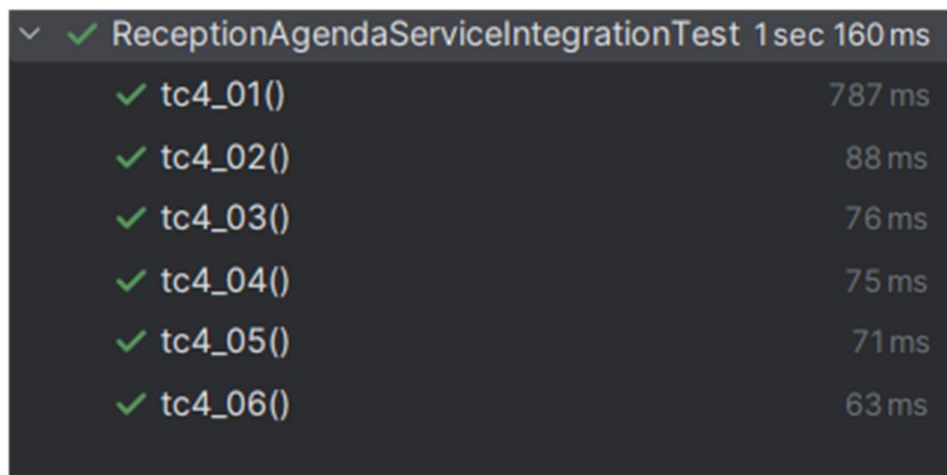


A screenshot of a JUnit test runner interface showing the results for the class ReceptionAgendaServiceTest. The header indicates the class name, package (it.astromark.agenda.r), and total execution time (1 sec 43 ms). Below, six individual test methods are listed, each with a green checkmark indicating success and its execution time.

✓ ReceptionAgendaServiceTest (it.astromark.agenda.r 1 sec 43 ms)	
✓ tc4_01()	1 sec 15 ms
✓ tc4_02()	7 ms
✓ tc4_03()	6 ms
✓ tc4_04()	4 ms
✓ tc4_05()	6 ms
✓ tc4_06()	5 ms

Integration service con repository

Il test di integrazione di ReceptionAgendaServiceImpl è stato effettuato utilizzando come contesto quella dell'intera applicazione ed è stata assegnata un'autorità con ruolo "Parent", per il genitore che intende effettuare la prenotazione del ricevimento. I test sono stati effettuati con l'ausilio degli stessi framework e librerie citate per il TC1. I test di integrazione di TC4 sono, dopo le modifiche effettuati, "Passati".



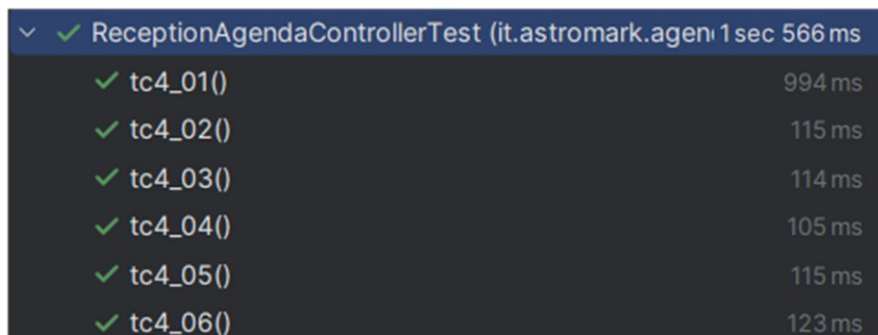
A screenshot of a JUnit test runner interface showing the results for the class ReceptionAgendaServiceIntegrationTest. The header indicates the class name and total execution time (1 sec 160 ms). Below, six individual test methods are listed, each with a green checkmark indicating success and its execution time.

✓ ReceptionAgendaServiceIntegrationTest 1 sec 160 ms	
✓ tc4_01()	787 ms
✓ tc4_02()	88 ms
✓ tc4_03()	76 ms
✓ tc4_04()	75 ms
✓ tc4_05()	71 ms
✓ tc4_06()	63 ms

Integration controller

Per il test dell'endpoint `/api/agenda/reception/timeslot/{id}/book` di `ReceptionAgendaController`, utilizzando il metodo HTTP POST, è stata effettuata una richiesta di autenticazione. Il token ottenuto è stato utilizzato come valore da inserire nel header "Authentication" per le successive richieste HTTP POST all'endpoint per la creazione di una nuova prenotazione.

Tutti i test sono stati superati con successo.



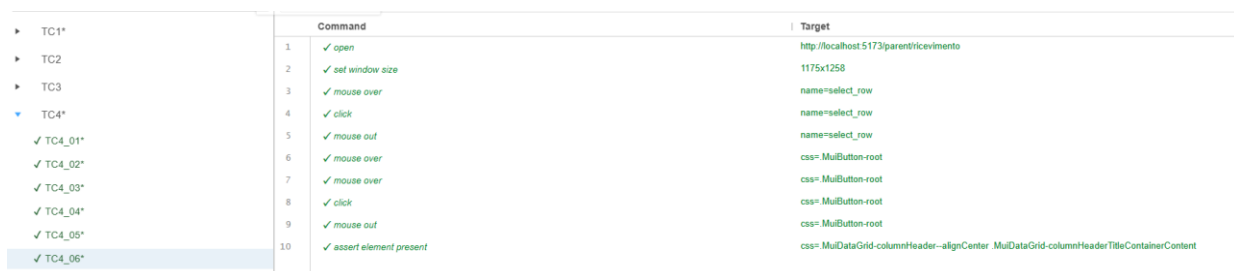
A screenshot of a test runner interface showing the results for `ReceptionAgendaControllerTest`. The test suite is marked as successful with a green checkmark and a duration of 1 sec 566 ms. Below the suite name, six individual test cases are listed, each with a green checkmark and its execution time in milliseconds.

Test Case	Duration
tc4_01()	994 ms
tc4_02()	115 ms
tc4_03()	114 ms
tc4_04()	105 ms
tc4_05()	115 ms
tc4_06()	123 ms

Test di Sistema

I test di sistema della funzionalità di prenotazione del ricevimento sono stati realizzati tramite Selenium IDE. La pagina oggetto di test è `/parent/ricevimento`, le asserzioni sono state effettuate sulla presenza di specifici elementi HTML che dovrebbero essere visualizzati in caso di errore.

Di seguito sono riportati gli screenshot dell'esecuzione della suite TC4 superati con successo.



A screenshot of the Selenium IDE interface showing a test suite named TC4*. The suite is expanded, revealing a list of test cases (TC4_01* to TC4_06*) and a detailed view of the commands and targets for each step.

Command	Target
1. open	http://localhost:5173/parent/ricevimento
2. set window size	1175x1258
3. mouse over	name=select_row
4. click	name=select_row
5. mouse out	name=select_row
6. mouse over	css= .MuiButton-root
7. mouse over	css= .MuiButton-root
8. click	css= .MuiButton-root
9. mouse out	css= .MuiButton-root
10. assert element present	css= .MuiDataGrid-columnHeader-alignCenter .MuiDataGrid-columnHeaderTitleContainerContent

4. Glossario

Termine	Descrizione
Test unitario	Verifica la correttezza di una singola unità di codice (es. una funzione o un metodo).
Test di integrazione	Verifica l'interazione tra diversi componenti di un sistema.
Test di sistema	Verifica il funzionamento completo del sistema.
Caso di test	Un insieme di condizioni o variabili usate per valutare un sistema.
Suite di test	Una raccolta di casi di test.
Scenario di test	Una sequenza di azioni eseguite per verificare una specifica funzionalità.
JUnit 5	Framework Java per la creazione e l'esecuzione di test unitari.
Mockito	Framework Java per la creazione di oggetti fittizi (mock) durante i test.
Testcontainers	Libreria Java che fornisce contenitori leggeri e usa e getta di database, browser e altre dipendenze per i test.
Faker	Libreria per generare dati fittizi.
Selenium IDE	Strumento per registrare e riprodurre test di applicazioni web.
MockMvc	Framework di testing per Spring MVC.
GitHub Actions	Piattaforma per l'integrazione continua e la consegna continua.
Pull request	Richiesta di unire delle modifiche al codice sorgente principale.
Docker	Piattaforma per la creazione e l'esecuzione di contenitori.
PostgreSQL	Sistema di gestione di database relazionale open-source.
JWT (JSON Web Token)	Standard per la trasmissione sicura di informazioni tra parti sotto forma di oggetto JSON.
Copertura del codice	Misura che indica la porzione di codice effettivamente testata.