

SAD 371 ASSIGNMENT 2

08/10/2025



Team Members

JESSICA WIEHE 600242, KAELO MOLEBATSI 600697, ARNAN KRUGER 601326, HENDRIK SMITH 600519, MARIUS VAN DER WESTHUIZEN 601075

Contents

[Introduction 2](#_Toc210411194)

[Problem Definition (½ page) 3](#_Toc210411195)

[Requirements Definition 4](#_Toc210411196)

[Functional Requirements (10 Marks) 4](#_Toc210411197)

[Non-Functional Requirements (10 Marks) 6](#_Toc210411198)

[Use Cases 7](#_Toc210411199)

[Diagram (15 Marks) 7](#_Toc210411200)

[Detailed Use Case Descriptions (10 Marks) 8](#_Toc210411201)

[Activity Diagram (10 Marks) 9](#_Toc210411202)

[Conclusion 10](#_Toc210411203)

[GitHub Repository link: 10](#_Toc210411204)

# Introduction

At Belgium Campus, tutoring is an effective way of offering students extra support and closing learning gaps. Scheduling sessions is currently done manually and, as a result, is faced with many problems. Students and tutors rely on word of mouth or group chats, which lead to double bookings, missed sessions, and limited access to assistance. This project is designed to create an Online Tutoring Booking System to make the process easier and more dependable. Using this system, students will have the possibility to book, reschedule, or cancel tutoring sessions, while tutors can advertise their availability in one place. The Academic Office will also benefit from accessing records and reports of tutoring activity. With this system, scheduling will be more efficient, academic support will be more convenient, and the tutoring process in general will be easier for everyone involved.

# Problem Definition (½ page)

Belgium Campus students often struggle to coordinate tutoring sessions effectively. Word of mouth, group messages, or random messages are the main methods used to schedule most of the sessions. This approach leads to confusion, double booking, and lost sessions. It also hinders students from knowing which tutors are available at what time. Therefore, many students end up losing valuable study time while trying to arrange with tutors. Tutors also have the same problems. Without a central system, they have trouble keeping appointments straight and avoiding scheduling conflicts. The Academic Office also does not get much insight into how tutoring is being used and cannot easily monitor or track student progress. This disorganization reduces the overall benefit of tutoring as a learning tool. An Online Tutoring Booking System addresses these issues by presenting a single point of session management. Students will see tutor availability, book appointments, and adjust bookings without unwarranted waiting time. Tutors will see their schedules clearly and be able to update their availability in real time. The Academic Office will have accurate records and reports, making program management and student success easier.

# Requirements Definition

## Functional Requirements (10 Marks)

**Process-Oriented**

1. The system must allow **students** to:

* Register and log in using their Belgium Campus credentials.
* Search for available tutors by subject, course code, or name.
* Book a tutoring session from available timeslots.
* Reschedule or cancel booked sessions.
* Receive reminders and notifications about upcoming sessions.
* View booking history.

1. The system must allow **tutors** to:

* Register and log in using their Belgium Campus credentials.
* Publish their availability (dates, times, subjects).
* Update or remove availability.
* View all booked sessions.
* Receive notifications of new bookings, cancellations, or reschedules.

1. The system must allow the **Academic Office (Admins)** to:

* View reports of all tutoring activities (sessions scheduled, rescheduled, cancelled).
* Monitor tutor and student participation.
* Generate performance reports (e.g., number of sessions per subject).
* Manage user accounts (approve tutors, deactivate users, etc.).

1. The system must automatically:

* Prevent double booking of a tutor or student at the same timeslot.
* Send confirmation emails/notifications after bookings, cancellations, or changes.
* Sync session data in real-time across all users.

**Information-Oriented**

1. The system must store:

* **Student data**: name, student ID, contact information, booking history.
* **Tutor data**: name, tutor ID, subjects offered, availability schedule.
* **Session data**: date, time, subject, tutor assigned, student assigned, status (booked, cancelled, completed).
* **Notifications data**: timestamps, delivery method (email/SMS/app).
* **Reports data**: summaries of tutoring activities for the Academic Office.

1. The system must allow retrieval of:
2. Available tutors by subject and timeslot.
3. Student booking history (past and future sessions).
4. Tutor schedules and workload.
5. Reports summarizing tutoring activity across time ranges (weekly, monthly, semester).

## Non-Functional Requirements (10 Marks)

**Operational Requirements**

Cross-Platform Responsiveness:

* The system must run seamlessly on both desktop and mobile devices to accommodate students and tutors who access the system from various devices. The user interface should automatically adjust to different screen sizes without losing functionality or readability.

Integration with Belgium Campus Authentication (Single Sign-On):

* The system must integrate with the institution’s existing authentication infrastructure to allow users to log in using their Belgium Campus credentials. This ensures consistent user management, reduces the need for multiple passwords, and improves data security.

User Interface and Usability:

* The interface must be intuitive and easy to navigate, requiring minimal training. Icons, colour coding, and clear labels should guide users through booking, cancelling, or managing tutoring sessions efficiently.

Language Accessibility:

* The system should be available primarily in English to align with campus operations but must be designed to support multilingual capabilities in the future. This ensures accessibility for students from diverse linguistic backgrounds.

System Availability and Reliability:

* The system must maintain an uptime of at least 99% during academic periods to ensure uninterrupted access for students and tutors. Scheduled maintenance should occur outside peak study hours.

**Performance Requirements**

Concurrent User Support:

* The system must handle at least 500 concurrent users without performance degradation. This ensures that students can make bookings even during peak registration periods.

Response and Load Time:

* Page load times must remain under two (2) seconds to provide a smooth user experience. Real-time updates should occur when bookings are made or cancelled to maintain data accuracy.

Data Processing and Confirmation:

* Booking confirmations must be processed instantly and displayed to the user in real-time. The database should update availability dynamically to prevent double bookings.

Report Generation:

* Administrative reports—such as session statistics, tutor utilization, and booking frequency—should be generated within 10 seconds, even for large datasets.

Scalability:

* The system architecture should support scalability to accommodate growth in the student population and future expansion to other departments or campuses.

**Security Requirements**

Data Protection and Encryption:

* All sensitive data (student and tutor information, login credentials, and booking details) must be encrypted both at rest and in transit using industry-standard encryption protocols (e.g., TLS (Transport Layer Security) for transmission, AES (Advanced Encryption Standard) for storage).

Authentication and Authorization:

* Access must be controlled through the Belgium Campus authentication system. Students, tutors, and administrators must have role-based permissions restricting their access to authorized functions only.

Session Management:

* User sessions must time out after 15 minutes of inactivity to prevent unauthorized access on shared or public devices.

Data Backup and Recovery:

* Automatic backups must occur daily to ensure that booking and user data can be restored in the event of a system failure or cyber incident. Backup files should be securely stored off-site.

Audit and Monitoring:

* The system must maintain logs of all user actions (e.g., logins, bookings, cancellations) for security auditing and troubleshooting purposes.

**Cultural and Political Requirements**

Institutional Compliance:

* The system must adhere to Belgium Campus’s internal IT and academic policies, ensuring that system access aligns with institutional roles and responsibilities.

Legal and Data Privacy Compliance:

* All data handling must comply with the Protection of Personal Information Act (POPIA) of South Africa. User consent for data storage and use must be obtained where applicable.

Trust and Transparency:

* The system must build user trust by being transparent about data usage, implementing visible security measures, and ensuring reliable performance.

Academic Integrity and Role Verification:

* Only verified tutors approved by the Academic Office may appear in the system. This ensures compliance with academic rules and prevents unauthorized individuals from providing tutoring services.

Inclusivity and Accessibility:

* The system should be designed to meet accessibility standards, ensuring that users with visual or physical impairments can still interact with it effectively (e.g., through screen reader compatibility and clear visual contrast).

# Use Cases:

## Diagram:

A screenshot of a computer

AI-generated content may be incorrect.

**Actors**

1. A[Student] actor
2. B[Tutor] actor
3. C[Academic Office (Admin)] actor
4. SSO[(Belgium Campus SSO)] ext
5. NOTIF[(Notification Service)] ext

Student use cases:

* Login[Login]
* Search[Search Tutors]
* ViewAvail[View Availability]
* Book[Book Session]
* Reschedule[Reschedule Session]
* Cancel[Cancel Session]
* History[View Booking History]

Tutor use cases

* Publish[Publish Availability]
* UpdateAvail[Update/Remove Availability]
* ViewBookings[View Booked Sessions]

Admin use cases

* Reports[View/Generate Reports]
* ManageUsers[Manage User Accounts]

Common/internal

* CheckAvail<<Check Availability>>
* SendConf<<Send Confirmation>>

## Detailed Use Case Descriptions:

**Use Case 1:**

**Primary Actor:** Student

**Stakeholders & Interests:**

* Student (quickly find an available tutor/time)
* Tutor (no clashes; clear updates)
* Academic Office/Admin (accurate records, utilization reporting)

SAD 371 Assignment 1 Final

**Preconditions:**

* Student is authenticated via Belgium Campus SSO.

SAD 371 Milestone 2 Part A done

* Tutor availability exists for relevant subject/time.

SAD 371 Milestone 2 Part A done

**Postconditions (Success):**

* Session stored with status = “Booked”; double-booking prevented; confirmation sent.

SAD 371 Milestone 2 Part A done

**Main Success Scenario:**

1. Students log in (SSO).
2. Students search for tutors by subject/course code/name.
3. System displays tutors and available timeslots.
4. Student selects a timeslot and requests booking.
5. System checks availability and prevents double booking.
6. System creates booking records, updates availability in real time.
7. System sends confirmation to students and notifications to tutors.
8. Students can view booking in history

**Use Case 2:**

**Primary Actor:** Student

**Stakeholders & Interests:** Student (flexibility), Tutor (accurate calendar), Admin (data integrity for reports).

**Preconditions:**

* Existing booking in “Booked” state.
* Student authenticated.

**Postconditions (Success):**

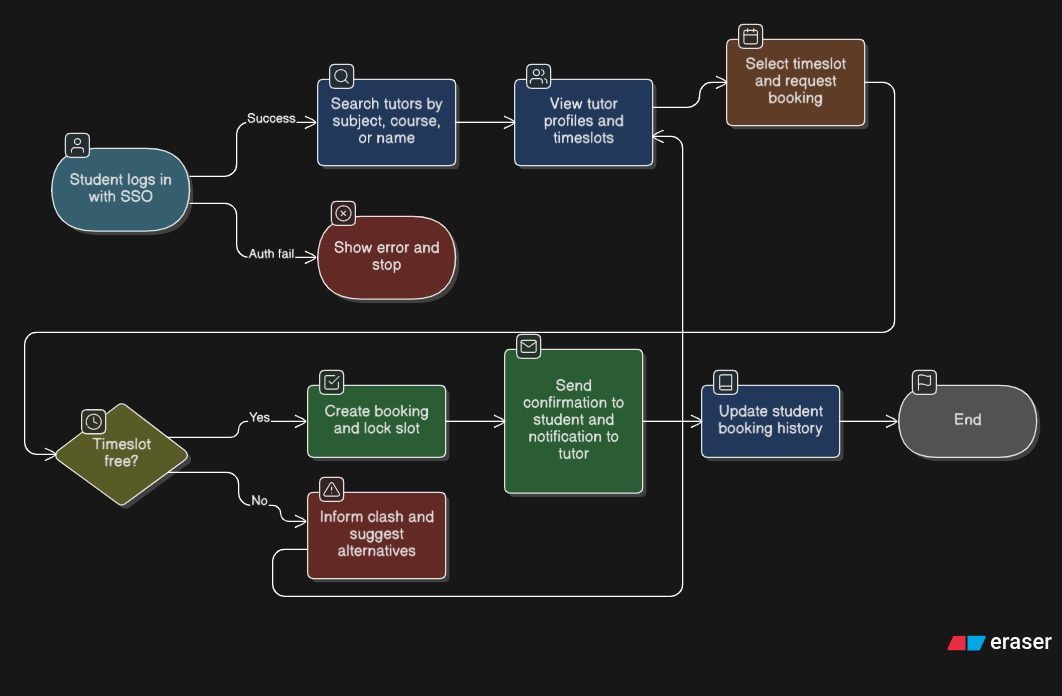
* Original slot released, new slot booked, notifications dispatched, audit trail kept.

**Main Success Scenario:**

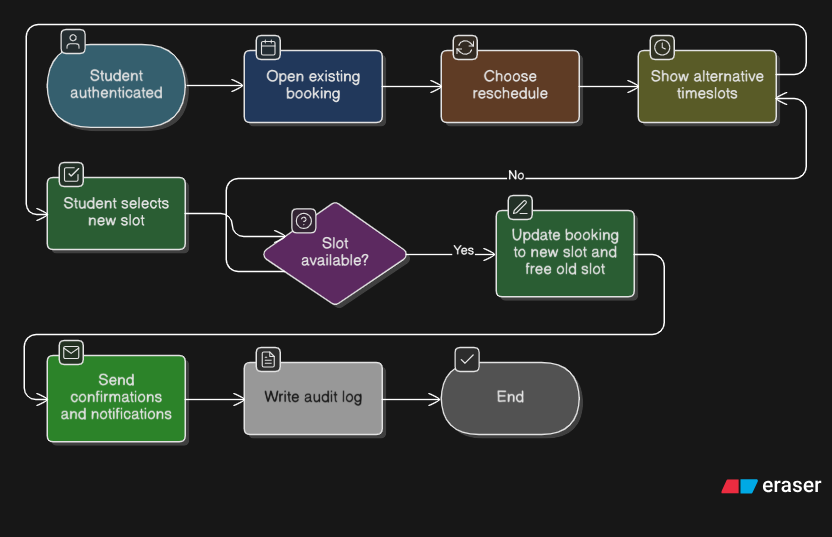
1. Student opens booking from history and selects “Reschedule.”
2. System shows tutor’s alternative timeslots.
3. Students pick a new slot.
4. System checks for conflicts and prevents double-booking.
5. System updates the session to the new date/time and frees the old slot, real-time sync.
6. System sends confirmations/notifications to both parties.
7. System logs the change for audit.

## Activity Diagram (10 Marks)

Book Session:



Reschedule Session:



# Conclusion

The Online Tutoring Booking System is designed to make life easier for both students and tutors. Instead of relying on messy, last-minute arrangements, it introduces a clear, digital way to schedule meetings by cutting down on double bookings and making learning support more accessible. Tutors can set their own availability with confidence, while the Academic Office gains reliable records and insights to better manage resources. Most importantly, students get a dependable way to connect with the help they need to thrive. This system tackles real challenges on campus and brings meaningful benefits to everyone it touches.

# GitHub Repository link:

<https://github.com/JessicaWiehe2003/SAD-371-Assignments>