Functional Requirements

- The application must let a user login via their Google account (3)
- All accounts must be authenticated
- The user must be able to customize their notifications
- The user must be able to schedule an event (2)
- The user must be able to view their agenda (1)
- A user must be able to schedule a meeting with two or more people (5)
- The application must be able to categorize an event automatically and intelligently (4)
- A user must be able to personalize, if necessary, the conditions of the meetings they schedule - Use Case #1
- A user must be able to accept or reject a meeting schedule - Use Case #1
- A user must be able to remove participants from their meeting.
- When scheduling an event that requires preparation, the user must be able to customize the time and slots for scheduling this preparation - Use Case #2
- The application must send notifications to users about events marked on their calendar
- The application must send warnings about goals that were not met
- The application must have a “to-do list” of events/tasks that the user wants to schedule. (Low Priority)
- The application must be able to suggest timeslots where to mark tasks from the user's to-do list (Low Priority)
- The user must be able to change the event tag, even though it is self-tagged by the application - ZeroShot Requirement
- The application must be able to suggest breaks in very long study/work sessions. (Low Priority)
- The user must be able to define a rest time.
- The user must be able to define the number of hours for work, free time, exercise time, rest periods, among other categories of events, that they intend to spend weekly.
- A user should be able to see their calendar according to their preferences (24h, 48h, day, week, month)
- The system should notify the user about the status of their week (busy, quiet)

- The system must be able to categorize events in different levels of priority according to their notifications (if they go off in the same day as the event or not, or if there's even a notification)
- The system must create "pending slots" while the user is waiting for every participant to choose one of the slots that they selected as available
- The user should be able to check how many people have selected a "pending slot"
- The user should be able to schedule new events in "pending slots" that have not been chosen by any participant
- The system should allow users that don't use it to select slots for joint meetings upon receiving an invitation (for example, via an interface)

Non-Functional Requirements

- The app should take no more than 5 seconds to find a compatible time among up to 100 people.
- When creating an event on the user's calendar, the application must be able to categorize the event in less than 2 seconds
- The error rate when timing an event must be less than 10%
- Compatibility - The application must be able to work correctly on both a desktop and a smartphone
- NLP model
- Web App
- Usability → The application must be easy to learn to use.
- Security - The application must follow a flow that minimizes the possibility of third parties redoing a user's schedule.
- Scalability - The application must be able to maintain good performance if the number of users increases and be easy to add new modules according to new features needed.
- The application's source code must follow good software development practices, such as modularity, readability and adequate comments, to facilitate maintenance and teamwork.