# Functional Specification

# Stefanini – Room Scheduler

Phase 1

**Release:** Draft

**Date:** 9th of October 2017

**Authors:** Raluca Nechita

# Report History

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision date** | **Author** | **Version** | **Summary of Changes** |
| 22 Sept 2017 | Raluca Nechita | V1.0 | Created First Draft |
| 2Oct 2017 | Raluca Nechita | V1.1 | Updated the draft |
| 2Oct 2017 | Raluca Nechita | V1.2 | Updated the draft with the feedback from the Kick-Off Meeting |
| 9 Oct 2017 | Irina Pandrea | V1.3 | Updated the draft with the requirements ID |
|  |  |  |  |
|  |  |  |  |

Contents

[Functional Specification 1](#_Toc494727729)

[Stefanini – Room Scheduler 1](#_Toc494727730)

[Report History 2](#_Toc494727731)

[Revision History 2](#_Toc494727732)

[Purpose 4](#_Toc494727733)

[Naming conventions 4](#_Toc494727734)

[Scope 5](#_Toc494727735)

[Global requirements 5](#_Toc494727736)

[Modules 6](#_Toc494727737)

[- Login section 6](#_Toc494727738)

[- Attendee section 6](#_Toc494727739)

[- Host Section 7](#_Toc494727740)

[- Admin section 7](#_Toc494727741)

# Purpose

This document provides outline functional specifications and requirements for the Room Scheduler web application, phase 1. Later phases will be determined at a later date, for which, a new document will be created.

It is designed to guide system development and design, including:

* Database structure
* Application architecture
* Site functionality
* Administrative toolset
* Use cases

# Naming conventions

* An **event** is something happening in an interval for a specific room
* **Attendee** is the default user role (the role that an AD user will automatically have it. This role is used by default for a newly created user). Attendee might be referred as “user” in this document.

The requirements set out here are ranked in **M**o**SC**o**W** order:

* **M** – Must Have
* **S** – Should Have
* **C** – Could Have
* **W** – Would like to have

# Scope

## Global requirements

* *1.01* The application will be localized **( C )**
* *1.02* Only active Stefanini domain users will have access to it **(M)**
* *1.03* Responsive Design **(S)**

The application will have **(M)**:

* *1.04* The application will have 3 user roles:
  + **Attendee** (the user that will book a room)
  + **Host** (the room host -> default will be “unknown”)
  + **Admin**
* *1.05* The application will have 3 types of events:
  + **Booked** (any user role has access to this event)
  + **Blocked** (only the Admin has access to this event)
* *1.06* An event can have one of the following statuses:
  + **Preset** (available only by the host or the admin)
  + **Absent** (available only by the host or the admin)
  + **Cancelled** (available for any role)
  + **Waiting** (available for the Host to go to Present or Absent. Available for the user for cancell)

*1.07* The Booked event status flows are:

* From waiting, an event can be marked as:
  + Cancelled
  + Present
  + Absent
* From Present, Absent, Cancelled, no other statuses will be available.

*1.08* The Blocked event has no statuses.

## Modules

The application will have the following modules and features:

### 2. Login section

*2.01* All active domain users will have access to this application **( C )**.

*2.02* There will be also the possibility to access the application using a username and a password **(M)**.

*2.03* (If only the login functionality will be available, the Admin->User Management and Roles sections will become **(M)**)

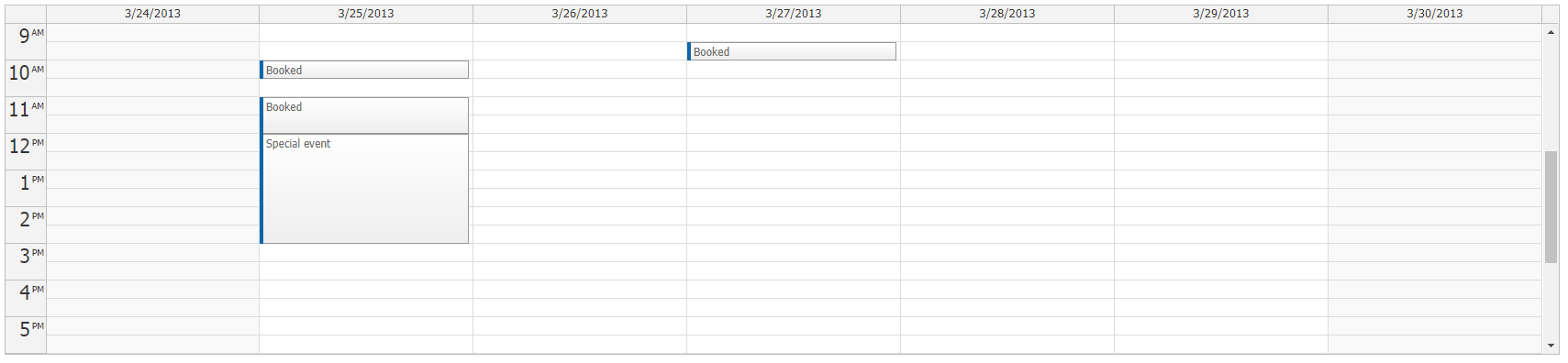
### 3. Attendee section

The user section will have:

* + - *3.01* A calendar, displaying the available time intervals and highlighting the busy ones. **(M)**

*3.02* The list of rooms will appear on the right corner as a filter. By default, the first room is loaded.

*3.03* This page will also have a legend, in which the hosts and their time interval will be displayed.



Example using: <https://javascript.daypilot.org/demo/lite/calendar/nextprevious.html>

*3.04* The already booked events will be displayed as “Booked” and highlighted in grey. No information about the user that booked will be displayed. Only the user that booked the room will see an “X” button on each **future** event that is assigned to itself.

*3.05* The events created by the user that is seeing the calendar will be highlighted differently

* + - * Past events will be highlighted in green if they happened = Present
      * Past events will be highlighted in red if they didn’t happened = Absent
      * Future events that will happen “today” in less than 30 minutes from “now” will be grey = Waiting
      * Future events will be blue = Waiting
        + Only for these events the user will see the “X” button / possibility to cancel the event. Once cancelled, the event history will be stored and the event will have the cancelled status for the user that cancelled it (please also store the cancel date and time).
      * *3.06* Cancelled events will not be displayed
    - *3.07* The calendar can have an additional filter, for hosts. By default all hosts are checked. If one of the hosts is deselected, the time interval in which the host is the owner will be disabled from the calendar **(W)**
    - Actions available on the calendar **(M)**:
      * *3.08* The possibility to book a free interval by click-> drag&drop **(M).**

Once stored, the event will be of type “booked” and will have the status “Waiting”. Please make sure to store the date and time when the user submitted the event.

*3.09* Restrictions:

* + - * + One person can book a maximum of 1h per day
        + One person can book a minimum of 0.5h per day
        + An event can be cancelled until 30 minutes before the actual start (value that should be configurable, for now, stored in the DB) – **( C )**
        + If a user had 3 events in which he/she was not present, the possibility to book a room will be redrawn for 15 day.
      * *3.10* Cancel/Delete an event **(M)**

### 4. Host Section

*4.01* A calendar page, as defined for Attendee. Accessible only by the host **(M)**.

*4.02* The host will be able to do the following actions:

* + For today’s past events, the host can mark an event as Present or Absent (while the event is still in “Waiting” status).
  + If the host failed to mark the today’s event and is already tomorrow, all yesterday’s event that are still in “Waiting” state will automatically be marked as Present

### 5. Admin section

* + *5.1* User management **(S)**
  + *5.2* Events management
    - Block an interval
  + *5.3* Roles Management **(W)**
  + *5.4* Change restrictions **(W)**
  + *5.5* Rooms Managementand Hosts management**( C )**
  + *5.6* Reports **(W)**