# Prenotalo: project high-level specification

Cecchetto A., d'Antonio S., Perrella C., Roccia E., Zerpa Ruiz J. E.

January 2024

In this document, we provide the initial idea and high-level specification of the project describing what are its objectives, what the system should do, what are the potential users and the more important use cases. The project deals with the development of a distributed system, made up of different micro-services, to manage bookings and related operations. Prenotalo is at its core a simple but comprehensive booking system.

**Project Glossary**

| **Term** | **Definition** |
|---|---|
| Consumer | Someone that requests or books a service offered by a Producer |
| Producer | Someone that provides services to Consumers |
| User | Platform user, so both Consumer and Producer |

## 1 Objectives

With objectives of the project we mean what the system aims to facilitate or solve. The system, Prenotalo, provides booking management functionalities also integrating additional services and utilities, like notification and analytics.

It facilitates interaction between consumers and producers. It aims to provide a simple and straightforward way for consumers to find, book, and manage their appointments or reservations. It helps producers manage their resources effectively.

## 2 Functionalities

For functionalities, we mean what the application should do in order to satisfy concretely the objectives. The main functionalities of the system will be:

- Provide a way to register user data.
- Provide a way for a producer to create new events.
- Provide a way for a consumer to look for available events.
- Provide a way for a consumer to book an event.
- Provide a way for a consumer to look for booked events.
- Provide a way for a consumer to cancel a booking.
- Provide a way for a producer to cancel events.
- Provide a way for a consumer to pay for an attended event.
- Send consumers notifications about booked events, e.g. reminders.
- Provide a way for a producer to create notifications about an event, e.g. broadcast messages.
- Collect surveys about completed events.
- Provide data analytics obtained by the system to producer.
- Provide a web application interface to use the system.
- Provide a mobile application interface to use the system.

## 3 Potential users

Such a system could potentially be used by any small business, merchant or self-employed practitioner. This booking system aims to be general enough to be useful any time there is a service offered in slots of time. Hence, some examples of potential users might be lawyers, doctors, hair salons, schools like driving schools, dancing schools, etc. On the other hand, clients of such businesses might use the system to book the offered services.

## 4 Sample use cases

To have in mind some actual use cases, you might think of lawyer offering hers/his services by appointment, providing a web interface to hers/his clients, in order to perform bookings, as well as a mobile application to manage reservations. Moreover, you can think of a driving school offering such service to book driving lessons, let the clients pay through the system, or sign their presence to the teaching classes, etc. Finally, any self-employed practitioner could use the system to manage hers/his time slots, see analytics about its most requested service, time frame, automatically collect surveys, and communicate with hers/his clients through the notification integration.

Sample use cases of the system are the following:

| **USE CASE: RequestEvent** |
|---|
| ID: 1 |
| Brief description:<br>a Consumer creates an event to put in the system. |
| Primary actors: Consumer |
| Secondary actors: Producer |
| Preconditions:<br>User needs to be registered. User needs to be authenticated. |
| Main flow:<br>1. Consumer selects “Request an event”.<br>2. Consumer finds the person that offers the event/service.<br>3. Consumer selects the type of service.<br>4. Consumer selects a time slot.<br>5. Booking system sends the event to the Producer.<br>6. Producer accepts/rejects.<br>7. Booking system notifies the consumer. |
| Postconditions: None |
| Alternative flows: Details |

| **USE CASE: CreateEvent** |
|---|
| ID: 2 |
| Brief description:<br>a Producer creates an event to put in the system. |
| Primary actors: Producer |
| Secondary actors: Consumer |
| Preconditions:<br>User needs to be registered. User needs to be authenticated. |
| Main flow:<br>1. Producer selects “Create an event”.<br>2. Producer selects the type of service.<br>3. Producer selects a time slot.<br>4. Booking System saves the event. |
| Postconditions: None |
| Alternative flows: Details |

| **USE CASE: LookForAvailableEvents** |
|---|
| ID: 3 |
| Brief description:<br>a Consumer looks for available events. |
| Primary actors: Consumer |
| Secondary actors: None |
| Preconditions:<br>User needs to be registered. User needs to be authenticated. Event is in the Booking System. |
| Main flow:<br>1. Consumer selects “Look for available events”.<br>2. Consumer selects a service.<br>3. The Booking Systems shows the available services. |
| Postconditions: None |
| Alternative flows: Details |

| **USE CASE: BookEvent** |
|---|
| ID: 4 |
| Brief description:<br>a Consumer books an event. |
| Primary actors: Consumer |
| Secondary actors: None |
| Preconditions:<br>User needs to be registered. User needs to be authenticated. Event is in the Booking System. |
| Main flow:<br>1. Consumer selects “Book an event”.<br>2. Consumer selects a service.<br>3. The Booking Systems shows the available services.<br>4. Consumer requests the service.<br>5. The Booking System sends the event to the Producer.<br>6. Producer accepts/rejects.<br>7. Booking system notifies the consumer. |
| Postconditions: None |
| Alternative flows: Details |

| **USE CASE: PayEevent** |
|---|
| ID: 5 |
| Brief description:<br>a Consumer pays for a booked event. |
| Primary actors: Consumer |
| Secondary actors: Payment authorization system |
| Preconditions:<br>User needs to be registered. User needs to be authenticated. Event is in the Booking System. |
| Main flow:<br>1. Consumer selects “Pay an event”.<br>2. Consumer selects an already booked event.<br>3. Consumer pays and Payment authorization system handles payment.<br>4. Payment authorization system notifies Booking System. |
| Postconditions: None |
| Alternative flows: Details |