Requirements for VIA Padel Club [VPC]

In case you find errors, or missing scenarios, or something is unclear, please let me know, and I will update the document.

In this project case various scenarios should send out an email to users. We will ignore the actual email-sending, and can just fake that initially. We will, however, design the system, so it is easy to swap out the functionality. Ideally this notification is done through "Domain Events", but that is outside the scope of the course. We will take a simpler approach, introduced in session 5.

Introduction to the format

The requirements combine user stories with a form of use case scenarios. It's probably a slightly different format than you are used to, but this format suits our needs well for this project.

The use cases are "small" in what they do, perhaps smaller than what would usually happen, but I wanted to show a different approach, than what you are used to.

The format is explained below. Each box contains a single **use case**.

A use case consists of:

- An ID for easy reference.
- A short description.
- A user story (re-arranged order).
- Success and failure scenarios, written on a specific format called Gherkin. Or just "given-when-then". These have IDs too, e.g. S1, S2, S3 for success scenarios, and F1, F2, F3, etc for failure scenarios. This is, again, for easy reference, and tracking of what is implemented. This means you mark UC4.S3 as completed, and so on.

The given part are the prerequisites for the scenario, what should be in place. They can quickly become elaborate, with a long list of things, but you will find most of them in this document are short. The hope is that various preconditions are checked by various scenarios. So instead of listing all of them for each scenario, I list the most relevant ones. It is assumed you setup the rest of the implicit preconditions correctly during your testing, as they should be fairly obvious.

NOTE: Some scenarios cannot be implemented right away. Some depends on other scenarios or other use cases.

Other scenarios depend on implementation techniques explained in session 5.

Sometimes example data is provided between pipes: | example data |.

Example use case format:

<Use case ID> - <Short description>

User story on the format:
In order to … [business value first]
As a … [some user type]
I want to … [some feature]

[Optional comments]

Success scenarios

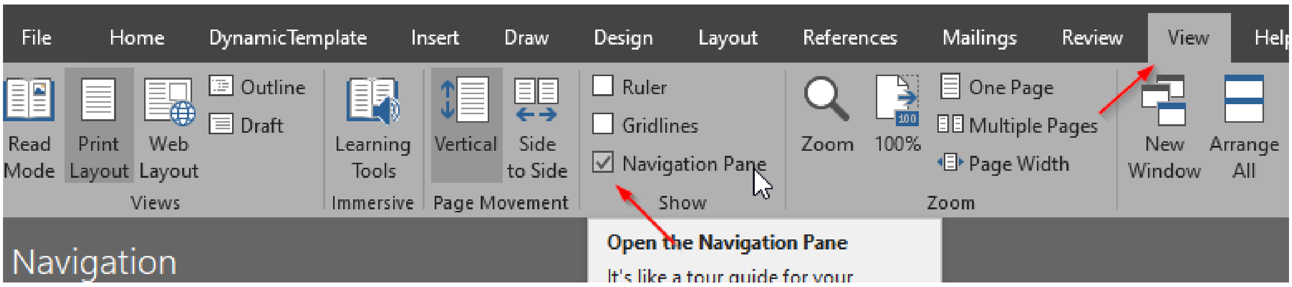
S1 **Given** … ***[And …]* ← *this means optional*** **When …** **Then** … ***[And …]***	**S2** **Given** … ***[And …]*** **When …** **Then** … ***[And …]***

Failure scenarios

F1 – Short title describing scenario **Given** … ***[And …]*** **When …** **Then** … ***[And …]***	**F2** **Given** … ***[And …]*** **When …** **Then** … ***[And …]***
F3 **Given** … ***[And …]*** **When …** **Then** … ***[And …]***	**F4** **Given** … ***[And …]*** **When …** **Then** … ***[And …]***

It can be a bit arbitrary when something is defined as two or more scenarios, or just one scenario. Sometimes I split it out in the hope of increasing clarity, and making it easier to define unit test cases.

I suggest making Word show the navigation panel on the left for easier overview of the document. You enable it like this:



Open the Navigation Pane
It's like a tour guide for your

Actors

Anonymous. This is someone who uses the system, but is not registered, nor logged in (we won't deal with logging in or out, so in this first iteration, we cannot distinguish between actors). Basically, *anonymous* can't do much with the system other than register.

Player. When *anonymous* registers, they become a Player. Players play padel, they can book padel courts, and they can be quarantined, or blocked, if they misbehave. This actor is registered in the system.

Manager is an *implicit* user, as in it does not really exist in the system. When *manager* is mentioned, we mean that VIA Padel Club (or some managing member of VPC) is performing some action. Eventually they will probably have some login credentials, and therefore exist in the system, but not in this first iteration. They don't *own* any data, so there is no need to store them in the system.

Authentication/authorization is not part of this course. You have seen approaches in previous courses, which could be applied, if you wish to expand upon the system later. There are alternative approaches too.

The manager is responsible for creating daily schedules, setting the correct information, and quarantining, blocking, unblocking players.

Requirements

The below requirements are ordered by importance. You are intended to implement these in order, as much as possible. Though, as mentioned above, some use cases depend on others, so you may have to temporarily skip a use case or just a scenario and come back again later.

Must have:

1) Manager creates daily schedule.
2) Manager updates start-time and -date, and end-time and -date of a daily schedule.
3) Manager adds available courts to a daily schedule.
4) Manager activates the daily schedule.
5) Anonymous registers a new account.
6) Player makes a booking.

Should have:

7) Player cancels a booking.
8) Manager removes available court from a daily schedule.
9) Manager quarantines player.
10) Manager blacklists player.
11) Manager un-blacklists player.
12) Manager adds VIP status to player.
13) Manager sets part of a daily schedule to VIP only.
14) VIP player books VIP only courts.
15) Manager deletes daily schedule.

Could have:

16) Player adds themselves to be notified in case of a booking cancellation, i.e. a queue system.
17) Player registers the results for a padel match.

Comment: Use cases in “Could have” have not been detailed. You are very welcome to implement them, but then I expect you to write use cases first, in the same format presented in this document.

I would love to see how you analyse the business logic for each feature! You are of course welcome to do this analysis even if you don’t plan on implementing the features, if you just want to practice.

Even though a “VIA login” is used, we do not authenticate the credentials against VIA’s authorization service. This can be considered a challenge exercise you can look at after the course is completed. Or you can hide the implementation behind a contract (cf. session 5) and make a fake implementation.

Use cases

ID: 1 – The manager creates a new daily schedule.

In order to make padel courts available to players
As a manager
I want to create a new daily schedule

Comment:

- Some properties will receive an initial default value when a schedule is created. You may want to postpone doing those success scenarios until you have implemented the use case related to that specific field.
  Or do some simple initial implementation.

Success scenarios

S1
Given -
When manager selects to create a daily schedule
Then a daily schedule is created with an ID
And the status is set to "*draft*"
And the list of available courts is empty
And the times are set to 15:00 and 22:00
And date is set to today

Failure scenarios

This should not be able to fail, except when the server is down. We do not describe such technical issues in these scenarios.

ID: 2 – The manager updates time and date on a daily schedule

In order to let players know which time and date bookings can be made
As a manager
I want to update the time and date on a daily schedule.

Note: For the "in the future/past" part in e.g. S2/F2, you need a Domain Contract, which is introduced in session 5. You can postpone implementing these specific rules.
You will also need a Domain Contract for the date conflict in F1.

Hint:

- You may look at the classes DateTime, DateOnly, and TimeOnly. The three can be converted between, e.g. TimeOnly.FromDateTime() will extract the time from a DateTime.
- You can do some basic math on DateTime, DateOnly, and TimeOnly.
- Make one or two methods for this.

Success scenarios

<table>
<tr>
<td><b>S1</b>
<b>Given</b> an existing daily schedule
<b>And</b> the chosen date is in the future
<b>And</b> the daily schedule is in <i>draft status</i>
<b>When</b> creator selects to set the date
<b>Then</b> the date of the daily schedule is updated
<b>And</b> the status remains in <i>draft</i></td>
<td><b>S2</b>
<b>Given</b> an existing daily schedule
<b>And</b> the daily schedule is in <i>draft status</i>
<b>And</b> the selected start time is before the end time
<b>And</b> the time between start and end is 60 minutes or more
<b>When</b> creator selects to set the date and time interval of a daily schedule
<code>| 10:00 to 14:00 |</code>
<code>| 08:00 to 16:00 |</code>
<b>Then</b> the times of the daily schedule are updated
<b>And</b> the status remains in <i>draft</i></td>
</tr>
</table>

Failure scenarios

F1 – No schedule found **Given** a date **And** no schedule has this date **When** the manager selects to set date and time interval of a daily schedule **Then** a failure message is provided explaining no daily schedule was found with this ID	**F2 – Date in the past** **Given** an existing daily schedule **And** the selected date is in the past **When** creator selects to set date and time interval of a daily schedule **Then** a failure message is returned explaining a daily schedule cannot be created in the past

F3 – Incorrect times

Given an existing daily schedule

When creator selects to set the time interval of a daily schedule

And the chosen end time is before the start time

```
| 10:00 to 09:00 |
| 08:00 to 07:59 |
| 12:00 to 12:00 |
```

Then a failure message is returned explaining that the end time must be after the start time

F4 – Too short time interval

Given an existing daily schedule

And the interval between start and end time is less than an hour

When creator selects to set date and time interval of a daily schedule

```
| 10:00 to 10:30 |
| 08:00 to 08:59 |
| 12:00 to 12:00 |
| 12:01 to 13:00 |
```

Then a failure message is returned explaining that the time interval must span 60 minutes or more

F5 – Schedule is active

Given an existing daily schedule

And the status of the schedule is *active*

When creator selects to set date and/or time interval of a daily schedule

Then a failure message is returned explaining an active daily schedule cannot modified, only deleted

F6 – Incorrect minutes

Given an existing daily schedule

When creator selects to set the times with the minutes not being :00 or :30

```
| 10:01 to 10:30 |
| 08:00 to 11:59 |
| 12:17 to 15:00 |
| 10:00 to 13:43 |
| 10:00 to 13:29 |
| 10:31 to 13:00 |
```

Then a failure message is returned explaining the minutes of the times must be half or whole hours.

ID: 3 – The manager adds available courts to a daily schedule

In order to let the players book a court
As a manager
I want to add courts to a daily schedule.

Success scenarios

S1 – General success

Given an existing daily schedule
And the status is *draft* or *active*
And the daily schedule is in the future
And the court name is on correct format:

- Example: S1, S2, D1, D2
- The court name must start with either the letter "s", "S", "d", or "D".
- The court name must end with a number: 1, 2, 3, 4, etc.
- The court number is between 1 and 10.
- The length of the court name is 2 or 3 characters.

When the manager selects to add a court to the daily schedule
Then the court is added to the list of available courts for this specific daily schedule
And the court name is capitalized, e.g. "s" is changed to "S"

S2 - First court added

Given an existing daily schedule
And there are currently no courts added
When the manager selects to add a court to the daily schedule
Then the court is added to the list of available courts for this specific daily schedule

S3 - Add with existing courts of different names

Given an existing daily schedule
And there is currently an existing court

```
| Existing court: D1. Added court: D2       |
| Existing courts: D1, D2. Added court: S1  |
```

When the manager selects to add a court to the daily schedule
Then the court is added to the list of available courts for this specific daily schedule

Failure scenarios

F1 – Past schedule

Given an existing daily schedule
And the daily schedule is in the past
When manager selects to add a court
Then a failure message is returned explaining that past schedules cannot be updated

F2 – Invalid starting letter

Given that the selected court name does not start with a valid letter (see S1)
When manager selects to add the court
Then a failure message is returned explaining the court naming rules

F3 – Deleted schedule **Given** an existing daily schedule **And** the daily schedule is deleted **When** manager selects to add a court **Then** a failure message is returned explaining that deleted schedules cannot be updated	**F4 – Invalid ending number** **Given** that the selected court name does not end with a valid number (see S1) **When** manager selects to add the court **Then** a failure message is returned explaining the court naming rules
F5 – Invalid name **Given** that the selected court name is not a valid length (see S1) **When** manager selects to add the court **Then** a failure message is returned explaining the court naming rules	**F6 – Schedule not found** **Given** a date **And** no schedule has this date **When** the manager selects to add the court **Then** a failure message is provided explaining no daily schedule was found with this ID
F7 – Court already exists **Given** an existing daily schedule **And** there is already a court with a given name in this schedule **When** the manager selects to add a court with the already taken name **Then** the request is rejected, with a message explaining the court is already added to the daily schedule.	

ID: 4 – The manager activates the daily schedule

In order to let players place bookings
As a manager
I want to make the daily schedule active, and thereby ready for bookings

Success scenarios

S1
Given an existing daily schedule
And there are 1 or more padel courts in the daily schedule
And the date and time of the daily schedule is in the future
When manager selects to activate the daily schedule
Then the status of the daily schedule is *active*

Failure scenarios

F1 – Missing courts **Given** an existing daily schedule **And** there are 0 padel courts in the daily schedule **When** manager selects to activate the daily schedule **Then** a failure message is returned explaining a daily schedule without and padel courts cannot be activated	**F2 – Schedule is in the past** **Given** an existing daily schedule **And** the start time of daily schedule is in the past **When** manager selects to activate the daily schedule **Then** a failure message is returned explaining a daily schedule with a past start time cannot be activated
F3 – Schedule is deleted **Given** an existing *deleted* daily schedule **When** manager selects to activate the daily schedule **Then** a failure message is returned explaining a deleted daily schedule cannot be activated	**F4 – Schedule not found** **Given** a date **And** no schedule has this date **When** the manager selects to activate the daily schedule **Then** a failure message is provided explaining no daily schedule was found with this ID
F5 – Date conflict **Given** an existing daily schedule **And** another *active* schedule exists with the same date **When** manager selects to activate the daily schedule **Then** a failure message is returned explaining that another active daily schedule has the same date	**F6 – Schedule is already active** **Given** an existing daily schedule **And** the schedule is already active **When** manager selects to activate the daily schedule **Then** a failure message is returned explaining that the schedule is already active

ID: 5 – An anonymous user (aka Anon) registers a new Player account

In order to properly use the platform
As an anonymous user (aka. Anon)
I want to register as a Player

Comment:

- I realize last names can come in many forms, like McGregor, O'Conner, De Santos, or Asian letters, etc. We'll stick to a simpler format.
- We do not store a profile picture. We store a URI to some picture online somewhere.
  Like this unknown-person-silhouette-whith-blue-tie.jpg (2048×2048) (istockphoto.com)
  There is a Uri class already in C#, you could look at.
  The Uri class will seemingly accept any string, as long as it is not empty. So, we won't care too much about correct URI, that can become too comprehensive to implement the rules ourselves. Or, consider it a challenge.
- Some scenarios could have been combined, I have split them in an attempt to make unit test cases easier.

Success scenarios

S1

Given via-email, first name, last name, and a profile picture URI
And email ends with "@via.dk"
And email is in correct email format
| <text1>@<text2>.<text3> |
And email:text1 is 3, 4 or 6 (inclusive) characters long
And email:text1 is either:

- 3 or 4 uppercase/lowercase English letters

  or
- 6 digits from 0 to 9

```
| trmo@via.dk   |
| jknr@via.dk   |
| iha@via.dk    |
| oih@via.dk    |
| 123456@via.dk |
| 987674@via.dk |
```

And first name
- is between 2 and 25 letters (a-z)
- contains only letters, i.e. no symbols, or spaces

And last name
- is between 2 and 25 letters (a-z)
- contains only letters, i.e. no symbols, or spaces

And the URI has correct format
When Anon chooses to register
Then a new account is created, with an ID and the above values
And the first letter of first name is capitalized
And the non-first letters of first name are lower-case
And the first letter of last name is capitalized
And the non-first letters of last name are lower-case
And the email is in all lower-case

<table>
<tr><th colspan="2">Failure scenarios</th></tr>
<tr>
<td><b>F1 – Incorrect email domain</b>
<b><u>Given</u></b> email
<b>And</b> email does not end with “@via.dk”
<b><u>When</u></b> Anon chooses to register with otherwise valid values
| trmo@gmail.com |
| trmo@viauc.dk |
| trmo@via.com |
<b><u>Then</u></b> the request is rejected, an error message is provided saying only people with a VIA mail can register</td>
<td><b>F2 – Incorrect email format</b>
<b><u>Given</u></b> email
<b>And</b> email is not in correct format (see S1)
| @via.dk |
| trmo@via |
| trmo@viadk |
| trmo@.dk |
| trmo.via.dk |
| trmoviadk |
<b><u>When</u></b> Anon chooses to register with otherwise valid values
<b><u>Then</u></b> the request is rejected, an error message is provided explaining the problem with the email</td>
</tr>
<tr>
<td><b>F3 – Email is empty</b>
<b><u>Given</u></b> email
<b>And</b> email is empty
| null |
| ““ |
| “ “ |
<b><u>When</u></b> Anon chooses to register with otherwise valid values
<b><u>Then</u></b> the request is rejected, an error message is provided explaining the problem with the email</td>
<td><b>F4 – Invalid image URI</b>
<b><u>Given</u></b> profile picture URI
<b>And</b> the URI is not a valid format (just check null or empty string)
<b><u>When</u></b> Anon chooses to register with otherwise valid values
<b><u>Then</u></b> the request is rejected, an error message is provided explaining url has incorrect format</td>
</tr>
<tr>
<td><b>F5 – First name is invalid</b>
<b><u>Given</u></b> first name
<b>And</b> first name is invalid (see S1)
<b><u>When</u></b> Anon chooses to register with otherwise valid values
<b><u>Then</u></b> the request is rejected, an error message is provided explaining the rules for first name</td>
<td><b>F6 – Last name is invalid</b>
<b><u>Given</u></b> last name
<b>And</b> last name is invalid (see S1)
<b><u>When</u></b> Anon chooses to register with otherwise valid values
<b><u>Then</u></b> the request is rejected, an error message is provided explaining the rules for last name</td>
</tr>
<tr>
<td><b>F7 – Email taken</b>
<b><u>Given</u></b> email
<b>And</b> the email is already registered
<b><u>When</u></b> Anon chooses to register with otherwise valid values
<b><u>Then</u></b> the request is rejected, an error message is provided explaining the email is already registered.</td>
<td></td>
</tr>
</table>

ID: 6 – The player makes a booking

In order to play padel tennis
As a player
I want to book a padel court

Notes

- A booking is minimum 1 hour.
- A booking is maximum 3 hours.
- A player cannot make a booking, if it leaves a hole in the schedule of less than 1 hour.
- The booking must be in increments of 30 minutes, e.g. 1h, 1½h, 2h, 2½h, etc.
- A booking must start on either half or whole hour, e.g. 14:00, 14:30. I.e. 14:15 is not allowed.
- There can be no overlaps in bookings of a padel court. E.g. 14:00-16:00 and 15:00-17:00 would create an overlap where two players expect to have reserved the court at the same time from 15:00 to 16:00.
- A player can have two bookings on the same date
- Parts of the schedule can be marked VIP only, and only players with VIP status can book courts overlapping with this VIP time span (UC13 and 14)
- The time duration can be indicated with.
    - Start and end times, e.g. 14:00 to 16:00.
    - Start time and duration, e.g. 14:00 and 1.5 hour.
    - Or maybe something else, it is up to you.

Success scenarios

S1
Given an existing Daily Schedule
And there is an available court
And the schedule is active
And the Player chooses valid time and duration
When Player selects to book this Court
Then the Court is marked booked by the Player in the given time interval

S2 - There are other bookings on the same court
Given an existing Daily Schedule
And the schedule is active
And there is a court
And there is one or more existing bookings on *the same court*
And the player chooses valid time and duration

```
|  Schedule: 09.00-21.00, court: D1                         |

|  Existing booking: 12.00-14.00                            |
|  New valid bookings: 11.00-12.00, 14.00-16.00, 09.00-12.00  |

|  Existing bookings: 11.00-13.00 and 17.00-18.00           |
|  New valid bookings: 09.00-11.00, 13.00-14.00, 15.00-17.00, |
|  18.00-20.00, …                                           |
```

When the Player selects to book this court
Then the Court is marked as booked by the Player in the given time interval

S3 - There is a time-overlapping booking on a different court

Given an existing Daily Schedule
And the schedule is active
And there are two or more courts
And there is one or more existing bookings on *another court*
And the player chooses valid time and duration

```
|  Schedule: 09.00-21.00, courts: D1, D2                            |

|  Existing booking: 12.00-14.00 on court D1                        |
|  New valid bookings on court D2:                                  |
|  11.00-12.00, 11.00-13.00, 13.30-15.00, 14.00-16.00               |
```

When the Player selects to book this court
Then the Court is marked as booked by the Player in the given time interval

Failure scenarios

F1 – schedule is deleted **Given** an existing Daily Schedule **And** the Daily Schedule is *deleted* **When** Player selects to book a Court **Then** the request is rejected, with a message explaining that courts cannot be booked if the Daily Schedule is not active	**F2 - Schedule is draft** **Given** an existing Daily Schedule **And** the Daily Schedule is *draft* **When** Player selects to book a Court **Then** the request is rejected, with a message explaining that courts cannot be booked if the Daily Schedule is not active
F3 - Schedule not found **Given** a date **And** no schedule has this date **When** a player selects to book a Court **Then** the request is rejected, with a message explaining that there was no daily schedule on this date	**F4 – Court not found** **Given** an existing Daily Schedule **And** the Daily Schedule is active **And** Player chooses valid time and duration **And** there are one or more Courts **When** Player selects to book a Court **And** that court does not exist in the daily schedule **Then** the request is rejected, with a message explaining that the court was not found
F5 – Booking start time before schedule start time **Given** an existing Daily Schedule **When** a Player selects to book a Court **And** the booking start time is before the schedule start time **Then** the request is rejected, with a message explaining the error	**F6 - Booking end time before schedule start time** **Given** an existing Daily Schedule **When** a Player selects to book a Court **And** the booking end time is before the schedule start time **Then** the request is rejected, with a message explaining the error

F7 - Booking start time after schedule end time

Given an existing Daily Schedule

When a Player selects to book a Court

And the booking start time is after the schedule start time

Then the request is rejected, with a message explaining the error

F8 - Booking end time after schedule end time

Given an existing Daily Schedule

When a Player selects to book a Court

And the booking end time is after the schedule start time

Then the request is rejected, with a message explaining the error

F9 - Incorrect start or end time format

Given an existing Daily Schedule

When a Player selects to book a Court

And the booking start or end time has an incorrect minute:

```
|  start or end: 14.01  |
|  start or end: 14.59  |
|  start or end: 14.15  |
|  start or end: 14.29  |
|  start or end: 14.31  |
|  start or end: 14.45  |
```

Then the request is rejected, with a message explaining the minutes must be 00 or 30

F10 - Booking too short

When a Player selects to book a Court

And the booking duration is less than one hour

```
|  14.00 - 14.30  |
|  14.00 - 14.00  |
|  14.00 - 14.59  |
```

(this may be caught by F9)

Then the request is rejected, with a message explaining a booking must be one hour or longer

F11 - Booking overlaps with other booking

Given an existing Daily Schedule

And chosen court

And there are other bookings for that court

And the player has selected a time span, which overlaps with one of the other bookings for the same court

```
|  Existing booking: 14-16  |
|  Selected booking: 14-16  |
```

When the player selects to book the court

Then the request is rejected, with a message explaining the court is not available in the selected time span

F12 - Booking too long

When a Player selects to book a Court

And the booking duration is more than three hours

```
|  14.00 - 17.30  |
|  14.00 - 18.00  |
|  14.00 - 17.01  |
```

(this may be caught by F9)

Then the request is rejected, with a message explaining a booking must be one hour or longer

F13 - Player is quarantined (UC9)

Given a player

And the player is currently quarantined

And the selected date of booking is before the quarantine has ended

When the player selects to book a court

Then the request is rejected, with a message explaining that the player cannot book courts on dates where they are quarantined

F14 - Player is blacklisted (UC10)

Given a player

And the player is blacklisted

When the player selects to book a court

Then the request is rejected, with a message explaining that the player cannot book courts when they are blacklisted

F15 - The booking overlaps with VIP time span (see UC12)

Given a daily schedule with courts
  And there is a VIP time span
  And the player is not marked as VIP (UC12)
  And the selected booking time span overlaps with the VIP time span

VIP time in the beginning of the schedule

```
|  Schedule Time: 9.00-21.00. VIP time: 9.00-11.00   |
|  Invalid booking times:                            |
|  9.00-10.00, 9.00-11.00, 10.30-13.00, 10.00-11.00  |
```

VIP time in the middle of the schedule

```
|  Schedule Time: 9.00-21.00. VIP time: 12.00-14.00  |
|  Invalid booking times:                            |
|  10.00-12.30, 12.00-13.00, 12.00-14.00, 12.00-14.30|
|  12.30-14.00, 12.30-13.30, 12.30-14.30, 13.30-14.30|
```

VIP time in the end of the schedule

```
|  Schedule Time: 9.00-21.00. VIP time: 19.00-21.00  |
|  Invalid booking times:                            |
|  18.00-19.30, 19.00-20.00, 19.00-21.00, 19.30-20.30|
|  18.00-21.00                                       |
```

When the player selects to book a court
Then the request is rejected, with a message explaining non-VIP players cannot place bookings, which overlaps the VIP time.

F16 - Player not found

Given an email supposedly for a player
  And no player with that email is registered
When a booking is requested
Then the request is rejected, saying no player was found with the provided email

F17 - Player already has booking on same date

Given an existing daily schedule
  And an email
  And the has a booking in this daily schedule
When the player makes another booking on this date
Then the request is rejected, with a message explaining a player can have a maximum of one booking per day.

F18 - Booking leaves a hole less than 1 hour

Given a daily schedule
  And the daily schedule may have other bookings on the chosen court
  And the booking would leave a gap in the schedule for this court of less than one hour

```
|  Schedule Time: 09-21. Court: D1. Existing booking: 14.00-16.00  |
|  Invalid booking times:                                          |
|  09.30-11.00, 19.00-20.30, 12.00-13.30, 16.30-18.00              |
```

When the player selects to book a court
Then the request is rejected, with a message explaining a booking may not leave gaps that are less than one hour.

ID: 7 – The player cancels a booking

In order to indicate I don’t need a booked padel court anyway
As a player
I want to cancel my booking

Notes

- Booking cannot be cancelled if the booking start time is less than an hour away.
- Email notifications are described in an independent scenario, as it is something which can be “added later”. It is secondary to the primary success scenario.

Success scenarios

S1 – Success cancellation
Given an existing booking
And the start time of the booking is more than one hour into the future

When the player selects to cancel the booking
Then the booking is cancelled

S2 – Queued players are notified
Given an existing schedule
And there is a booking on this schedule
And one or more players are queued for notifications
When the player selects to cancel their booking
Then the booking is cancelled, and the queued players receive a notification about the cancellation

S3 - Date before schedule, but time less than 1 hour
Given an existing schedule
And an existing booking
And the date of cancellation is before the schedule date
And the time of cancellation is less than 1 hour before the booking

```
|  Schedule date and time: 31-01-2025, 09.00-21.00   |
|  Booking time: 13.00-14.00                         |
|  Date and time of cancellation: 30-01-2025, 12.47  |
```

When the player selects to cancel the booking
Then the booking is cancelled

<table>
<tr><th colspan="2">Failure scenarios</th></tr>
<tr>
<td>**F1 – Booking is in the past**
Given an existing schedule
And an existing booking of a court
And the booking is in the past

| Date of schedule: 30-01-2025 |
| Time of booking: 12.00-14.00 |

| Invalid time of cancellation:|

| Date: 31-01-2025 |
| Times: 10.00, 11.00, 11.59 |

| Date: 30-01-2025 |
| Times: 12.00, 12.01, 13.00 |

When the player selects to cancel the booking
Then the request is rejected, with a message saying past bookings cannot be cancelled</td>
<td>**F2 – Cancellation is too late**
Given an existing schedule
And an existing booking of a court
And the booking is in the future
And the time of cancellation is less than an hour before the booking

| Date of schedule: 31-01-2025 |
| Time of booking: 11.00-14.00 |

| Invalid times of cancellation:|
| 31-01-2025, 10.01, 10.30, |
| 10.59, 11.00 |

| Date of schedule: 31-01-2025 |
| Time of booking: 11.30-14.00 |

| Invalid times of cancellation:|
| 31-01-2025, 10.31, 10.45, |
| 10.59, 11.30 |

When the player selects to cancel the booking
Then the request is rejected, with a message saying past bookings cannot be cancelled</td>
</tr>
<tr>
<td>**F3 – No booking found**
Given an existing schedule
And relevant booking information
And no matching booking exists
When the player selects to cancel a booking
Then the request is rejected, with a message saying that the no booking was found</td>
<td>**F4 – No schedule found**
Given relevant booking information
And no schedule matches
When the player selects to cancel a booking
Then the request is rejected, with a message saying that schedule on the requested date was found</td>
</tr>
<tr>
<td>**F4 – Player does not own booking**
Given an existing schedule
And relevant booking information
And the player does not own the booking
When the player selects to cancel the booking
Then the request is rejected, with a message saying that players can only cancel their own bookings</td>
<td></td>
</tr>
</table>

ID: 8 – The manager removes available court from a daily schedule

In order to not let players book padel courts, we can't use
As a manager
I want to remove a padel court from a daily schedule

Comment: S2 assumes we send out email updates. We don't necessarily need to actually send an email and can fake this with a technique introduced in session 5.

Note:

- You cannot remove a court from a schedule on the same date as the schedule, if there are bookings for this court later in the day.
- You can, however, remove the court, if all bookings are earlier in the day, than the time of removal.

Success scenarios

S1 **Given** an existing daily schedule **And** the event status is *draft* **And** a padel court name **And** the padel court exists in the daily schedule **When** manager selects to remove the padel court **Then** the padel court is removed from the daily schedule	**S2 – Email notification upon removal** **Given** an existing daily schedule **And** a padel court name **And** the event status is *active* **And** the padel court exists in the daily schedule **And** there is one or more bookings for the padel court to be removed **And** the date of removal is *before* the date of the schedule **When** manager selects to remove the padel court **Then** the padel court is removed from the daily schedule **And** an email is sent out to notify the relevant players that their booking has been cancelled
S3 - Only 1 court present **Given** an existing schedule **And** there is a single court added **When** the manager selects to remove the court **Then** the court is removed **And** there are no courts on the schedule	**S4 - Multiple courts present** **Given** an existing schedule **And** there are two or more courts added **When** the manager selects to remove a specific court **Then** the court is removed **And** the other court(s) are still present on the schedule

S5 – Bookings earlier in the day

Given an existing daily schedule
And a padel court name
And the event status is *active*
And the padel court exists in the daily schedule
And there is one or more bookings for the specific padel court
And the start time of all the bookings is *before* the time of removal
And the date of removal is the same date as the date of the schedule

```
| Date of schedule and removal:     |
| 31-01-2025                        |
| Times of bookings:                |
| 9.00-11.00, 12.30-14.00           |
| Valid times of removal:           |
| 12.31, 13.00, 14.00, 14.01, 15.00 |
```

When manager selects to remove the padel court
Then the court is removed

Failure scenarios

F1 – Schedule is in the past

Given an existing daily schedule
And the event status is *draft* or *active*
And a padel court name
And the daily schedule is in the past
When manager selects to remove the padel court
Then a failure message is returned explaining past daily schedules cannot be modified.

F2 – No padel court found

Given an existing daily schedule
And a padel court name
And the padel court name is not in the daily schedule
When manager selects to remove the padel court
Then a failure message is returned explaining the error.

F3 – Booking is ongoing

Given an existing daily schedule
And a padel court name
And the event status is *active*
And the padel court exists in the daily schedule
And there is one or more bookings for the specific padel court
And the time of removal overlaps with a booking, i.e. people are actively playing on the court
When manager selects to remove the padel court
Then a failure message is returned explaining active courts cannot be removed

F4 – No daily schedule found

Given an ID for a daily schedule
And no daily schedule with this ID exists
When manager selects to remove the padel court
Then a failure message is returned explaining that no matching daily schedules exists

F5 – Bookings later on the same day

Given an existing daily schedule

And a padel court name

And the event status is *active*

And the padel court exists in the daily schedule

And there is one or more bookings for the specific padel court

And the start time of the bookings is after the time of removal

And the time of removal is the same date as the date of the schedule

```
| Date of schedule and removal:   |
| 31-01-2025                      |
| Times of bookings:              |
| 9.00-11.00, 12.30-14.00         |
| Invalid times of removal:       |
| 8.59, 9.00, 11.00, 11.01, 12.29 |
| 12.30                           |
```

When manager selects to remove the padel court

Then a failure message is returned explaining that courts with bookings later on the same day cannot be removed

ID: 9 – The manager quarantines player

In order to punish players who do not show acceptable sportsmanship

As a manager

I want to quarantine a player, so they cannot book courts until the quarantine ends

Comment: S3 is potentially somewhat complicated, regarding how to retrieve the correct data. You will need to get the daily schedules, where the player has a booking. Are you just fetching all future schedules? Or does the player entity know which courts they have booked? Modelling this efficiently requires some deliberate thought.

Success scenarios

S1

<u>Given</u> an existing Player

And the current date

Quarantining date	Quarantine active until, including, date
20-01-2025	23-01-2025
28-01-2025	31-01-2025
30-01-2025	02-02-2025

<u>When</u> manager selects to quarantine the player

<u>Then</u> the player is quarantined

And the end date of the quarantine is set to three days into the future, relative to the time of quarantining

S2 – Player is already quarantined, add three days

<u>Given</u> an existing player

And the player is already quarantined

Currently quarantined until, including	Updated quarantine date
20-01-2025	23-01-2025
28-01-2025	31-01-2025
30-01-2025	02-02-2025

<u>When</u> the manager selects to quarantine the player

<u>Then</u> another 3 days is added to the quarantine period

S3 – All bookings during the quarantine are cancelled, with emails

<u>Given</u> an existing player

And the player has one or more bookings across one or more schedules

Quarantine date: 20-01-2025

Quarantine until, and including: 23-01-2025

Current bookings	Intact bookings
20-01-2025, 13.00-15.00	None
23-01-2025, 13.00-15.00	None
24-01-2025, 13.00-15.00	24-01-2025, 13.00-15.00
20-01-2025, 13.00-15.00 24-01-2025, 13.00-15.00	No 24-01-2025, 13.00-15.00
20-01-2025, 13.00-15.00 23-01-2025, 13.00-15.00 24-01-2025, 13.00-15.00	No No 24-01-2025, 13.00-15.00

<u>When</u> the manager selects to quarantine the player

<u>Then</u> the player is quarantined

And all bookings by this player during the quarantine period are cancelled

And all bookings *after* the quarantine end date are still intact

And any players in a queue for booking are notified

Failure scenarios	
F1 – Player not found I don’t have to spell this out again	**F2 – Player is already blacklisted** **Given** an existing player **And** the player is blacklisted (see UC10) **When** the manager selects to quarantine a player **Then** the request is rejected, with a message explaining the selected player is already blacklisted

ID: 10 – The manager blacklists a player

In order to punish a player who continuously violate our rules
As a manager
I want to blacklist a player, so they cannot book courts until unblacklisted

Comment: S3 is potentially somewhat complicated, regarding how to retrieve the correct data. You will need to get the daily schedules, where the player has a booking. Are you just fetching all future schedules? Or does the player entity know which courts they have booked? Modelling this efficiently requires some deliberate thought.

Note:

- Again, what you might consider a single success scenario is split into three. This is so you can do the basic first, and expand with extra functionality. And to hopefully better separate the unit tests.

Success scenarios

S1 **Given** an existing Player **When** manager selects to blacklist the player **Then** the player is marked as blacklisted	**S2 – Player's quarantine is removed** **Given** an existing Player **And** the player has an active quarantine **When** manager selects to blacklist the player **Then** the player is marked as blacklisted **And** the player's quarantine is cleared
S3 – Booked courts are cancelled, with emails **Given** an existing player **And** the player has bookings in the future **When** the manager selects to blacklist the player **Then** the player is marked as blacklisted **And** all future bookings are cancelled **And** any players in queue in any of the affected schedules are notified	

Failure scenarios

F1 – Player not found You know how this goes	**F2 – Player is already blacklisted** **Given** an existing player **And** the player is already blacklisted **When** the manager selects to blacklist the player **Then** the request is rejected, with a message explaining the selected player is already blacklisted

ID: 11 – The manager lifts the blacklisting of a player

In order to acknowledge that a player has repented their faulty ways (or they have provided a substantial donation)
As a manager
I want to remove the blacklisting of a blacklisted player

Comment: Regarding F2, you might consider doing nothing, when lifting a blacklisting of a player not currently blacklisted. I have chosen to let the manager know the blacklist-lifting did no work as expected. The manager may have selected the wrong player, and so the blacklisted player remains blacklisted. I consider this an error.

Success scenarios

S1 **Given** an existing Player **When** manager selects to blacklist the player **Then** the player is no longer blacklisted	

Failure scenarios

F1 – Player not found You know how this goes	**F2 – Player is not blacklisted** **Given** an existing player **And** the player is not currently blacklisted **When** the manager selects to blacklist the player **Then** the request is rejected, with a message explaining the selected player is not blacklisted

ID: 12 – The manager elevates a player to VIP status

In order to reward loyal, active, helpful players (or they have provided a substantial donation)
As a manager
I want to elevate a player to VIP status

Success scenarios

S1 **Given** an existing Player **When** manager selects to mark the player as VIP **Then** the player is marked as VIP	**S2** **Given** an existing Player **When** manager selects to mark the player as VIP **Then** the player is marked as VIP **And** the player is notified about the good news

Failure scenarios

F1 – Player not found You know how this goes	**F2 – Player is already VIP** **Given** an existing Player **And** the player is already marked as VIP **When** manager selects to mark the player as VIP **Then** the request is rejected, with a message explaining the player is already VIP. **And** no notification is sent to the player
F3 – Player is quarantined **Given** an existing player **And** the player is currently quarantined **When** the manager selects to mark the player as VIP **Then** the request is rejected, with a message explaining quarantined players cannot be elevated to VIPs	**F4 – Player is blacklisted** **Given** an existing player **And** the player blacklisted **When** the manager selects to mark the player as VIP **Then** the request is rejected, with a message explaining blacklisted players cannot be elevated to VIPs

ID: 13 – The manager sets part of the daily schedule as VIP only

In order to give priority to our VIP members
As a manager
I want to make part of a daily schedule VIP only

Comment:

- S1 should be easy enough, if you stick to just one allowed time span. If you want to include the other success scenarios, I think this will be a bit of a challenge. Consider yourself challenged.

Elaboration:

- This just means that only players marked as VIP can create bookings, which overlaps this VIP time span
- Multiple VIP time spans per daily schedule are allowed

Success scenarios

S1 – Zero pre-existing time spans
Given an existing daily schedule
And a time range, within the daily schedule
When the manager selects to mark this time range as VIP only
Then the daily schedule has a time range which is marked VIP only
And only VIP players can create bookings which overlaps this time range.
	S2 - Other time span exists, results in two separate
	Given an existing daily schedule
	And the daily schedule has an existing VIP time range
	And a new, separate time range
	`| Existing range: 12.00-14.00 |`
	`| Added range: 19.00-21.00    |`
	`| Result:                      |`
	`|   [12.00-14.00, 19.00-21.00 |`
	When the manager selects to mark this time range as VIP only
	Then the daily schedule has a new time range which is marked VIP only
	And the already existing time range is intact

S3 - One pre-existing time span, result in one time span

Given an existing daily schedule

> **And** the daily schedule has an existing VIP time range
>
> **And** a new, separate time range, which borders the existing

```
| Existing range: 12.00-14.00 |

| Added range: 14.00-15.00    |
| Result: 12.00-15.00         |

| Added range: 10.00-12.00    |
| Result: 10.00-14.00         |

| Added range: 10.00-12.30    |
| Result: 10.00-14.00         |

| Added range: 10.00-15.00    |
| Result: 10.00-15.00         |

| Added range: 12.30-15.00    |
| Result: 12.00-15.00         |

| Added range: 11.00-15.00    |
| Result: 11.00-15.00         |
```

When the manager selects to mark this time range as VIP only

Then the daily schedule has an updated time range which is marked VIP only

S4 - More pre-existing time spans, are combined as needed.

Given an existing daily schedule

> **And** the daily schedule has one more existing VIP time range(s)
>
> **And** a new, separate time range, which overlaps one or more of the existing

```
| Existing ranges:
|   11.00-13.00, 17.00-18.00

| Added range: 10.00-17.00
| Result: 10.00-18.00

| Added range: 10.00-17.00
| Result: 10.00-18.00

| Added range: 10.00-14.00
| Result: 10.00-14.00, 17.00-18.00

| Added range: 10.00-19.00
| Result: 10.00-19.00

| Added range: 17.00-19.00
| Result: 11.00-13.00, 17.00-19.00

| Added range: 13.00-17.00
| Result: 11.00-18.00
```

When the manager selects to mark this time range as VIP only

Then current VIP time ranges are updated accordingly

S5 – More pre-existing time spans

Define your own test data here

Failure scenarios

F1 – Schedule not found

As usual

F2 – Selected VIP time span already contains bookings

Given an existing daily schedule

> **And** the schedule contains bookings
>
> **And** one or more of these bookings overlap with the chosen VIP time span
>
> **And** one or more of these bookings are by non-VIP players

When the manager selects to add the chosen VIP time span

Then the request is rejected, with a message explaining that the chosen time span overlaps with existing bookings by non-VIP players

F3 – Chosen time span is outside of daily schedule time span

Given an existing daily schedule

And a chosen range for the VIP time span

And either:

- The VIP start time is before the schedule start time
- The VIP end time is before the schedule start time
- The VIP start time is after the schedule end time
- The VIP end time is after the schedule end time

(Some of the above may be caught by other rules)

When the manager selects to add the chosen VIP time span to the schedule

Then the request is rejected, with a with message explaining the error in the chosen VIP time span

F4 - Incorrect format of .00 or .30

Given an existing daily schedule

And a chosen range for the VIP time span

And the VIP start or end time does not end with .00 or .30

When the manager selects to add the chosen VIP time span to the schedule

Then the request is rejected, with a message explaining that VIP time spans must end with hole or half hours, i.e. the minutes must be .00 or .30

ID: 14 – VIP player books court

In order to give priority to VIP players
As a player with VIP status
I want to book a court in or across a time span which is marked VIP only

Note:

- Any one or more time span(s) in the daily schedule can be marked VIP. It can be in the beginning or the end. Or sometime in the middle. See UC13.

Dependency on UC6:

- Essentially the same rules apply as in UC6. It is just elaborated here in the specific case of a player with VIP status. You must still adhere to the rules in UC6, where it is relevant.
- A booking may:
  - Start at the same start time of the VIP time span
  - Start inside the VIP time span
  - Span across the entire VIP time span
  - End at the start or end of the VIP time span
  - Basically, any kind of overlap between booking and VIP time span is allowed

Success scenarios

S1
Given an existing daily schedule
And a VIP-player
And the daily schedule has a court
And the daily schedule has time span marked as VIP

| VIP time span: 14:00 – 15.30 |
| Booking time: 13:00 – 14:00 |
| Booking time: 13:00 – 14:30 |
| Booking time: 13:30 – 16:00 |
| Booking time: 15:30 – 17:00 |
| Booking time: 14:30 – 16:00 |

When the player books the court
Then the court is booked by the player

Failure scenarios

See failure scenarios of UC6

ID: 15 – The manager deletes a daily schedule

In order to remove a daily schedule in case the padel club changes their mind
As a manager
I want to delete a daily schedule

Note:

- We will do soft-deletion. The schedule will be marked deleted, but not actually removed from the database.
- Similarly to removing of padel courts, the manager is not allowed to delete a daily schedule on the same date as the daily schedule. This will be too short notice for the players, and is considered unfair.

Success scenarios

S1 – Schedule is active **Given** an existing daily schedule **And** the schedule is in the future **And** the schedule is *active* **When** manager selects to delete the schedule **Then** the schedule is marked as deleted **And** all bookings are cancelled **And** all courts are removed from the schedule **And** affected players are notified	**S2 – Schedule is draft** **Given** an existing daily schedule **And** the schedule is in the future or the same date as the date of deletion **And** the schedule is *draft* **When** manager selects to delete the schedule **Then** the schedule is marked as deleted **And** all courts are removed from the schedule

Failure scenarios

F1 – Schedule not found As usual	**F2 – Schedule is in the past** You can do this
F3 – Already deleted You can also do this	**F4 – Too late** **Given** an existing daily schedule **And** the schedule is *active* **And** the date of deletion is the same date as the schedule **When** the manager selects to delete the schedule **Then** the request is rejected, with a message explaining that a schedule cannot be deleted on the same day it is executed