RASD.docx

Main document changes and comments

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

16

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

15

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

18

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

17

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

19

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

18

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

20

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

19

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

20

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

19

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

20

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

19

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

23

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

22

Page i: Inserted	Marco Petri	08/12/2020 18:16:00

23

Page i: Deleted	Marco Petri	08/12/2020 18:16:00

22

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

23

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

22

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

24

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

23

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

26

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

25

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

28

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

27

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

28

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

27

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

29

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

28

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

29

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

28

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

30

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

29

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

30

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

29

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

31

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

30

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

31

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

30

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

32

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

31

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

32

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

31

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

33

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

32

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

33

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

32

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

33

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

32

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

34

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

33

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

34

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

33

Page i: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

35

Page i: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

34

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

35

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

34

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

36

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

35

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

36

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

35

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

36

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

35

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

38

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

37

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

38

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

38

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

38

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

38

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

40

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

40

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

40

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

40

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

39

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

41

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

40

Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

41

Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**
40		
Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**
42		
Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**
41		
Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**
42		
Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**
41		
Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**
49		
Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**
43		
Page ii: Inserted	**Marco Petri**	**08/12/2020 18:16:00**
50		
Page ii: Deleted	**Marco Petri**	**08/12/2020 18:16:00**
44		
Page 2: Formatted Table	**Marco Petri**	**06/12/2020 18:39:00**
Formatted Table		
Page 2: Inserted	**Marco Petri**	**06/12/2020 18:40:00**
	5	
Page 2: Deleted	**Marco Petri**	**06/12/2020 18:40:00**
	6	
Page 2: Inserted	**Marco Petri**	**06/12/2020 18:40:00**
	6	
Page 2: Deleted	**Marco Petri**	**06/12/2020 18:40:00**
	7	
Page 2: Inserted	**Marco Petri**	**06/12/2020 18:40:00**
	7	
Page 2: Deleted	**Marco Petri**	**06/12/2020 18:40:00**
	8	
Page 2: Inserted	**Marco Petri**	**06/12/2020 18:40:00**
	8	
Page 2: Deleted	**Marco Petri**	**06/12/2020 18:40:00**

9

Page 2: Inserted	Marco Petri	06/12/2020 18:40:00

9

Page 2: Deleted	Marco Petri	06/12/2020 18:40:00

10

Page 2: Inserted	Marco Petri	06/12/2020 18:40:00

0

Page 2: Deleted	Marco Petri	06/12/2020 18:40:00

1

Page 2: Inserted	Marco Petri	06/12/2020 18:39:00

1

Page 2: Deleted	Marco Petri	06/12/2020 18:39:00

2

Page 2: Inserted	Marco Petri	06/12/2020 18:39:00

2

Page 2: Deleted	Marco Petri	06/12/2020 18:39:00

3

Page 2: Inserted	Marco Petri	06/12/2020 18:39:00

3

Page 2: Deleted	Marco Petri	06/12/2020 18:39:00

4

Page 4: Commented [MP1]	Marco Petri	01/12/2020 15:22:00

ROSSI DOESN'T LIKE MACHINE PHENOMENA IN RASD

Page 7: Formatted	asus	03/12/2020 06:27:00

English (United Kingdom)

Page 7: Formatted	asus	03/12/2020 06:27:00

English (United Kingdom)

Page 7: Formatted	Marco Petri	02/12/2020 11:13:00

English (United Kingdom)

Page 7: Formatted	asus	03/12/2020 06:27:00

English (United Kingdom)

Page 7: Formatted	asus	03/12/2020 06:27:00

English (United Kingdom)

Page 7: Formatted	asus	03/12/2020 06:27:00

English (United Kingdom)

Page 8: Formatted	Marco Petri	06/12/2020 18:21:00

Space After: 12 pt

Page 8: Deleted **Marco Petri** **06/12/2020 18:21:00**

Page 8: Formatted **Marco Petri** **06/12/2020 18:22:00**

Keyword, Font: Not Bold, Italian (Italy)

Page 8: Formatted **Marco Petri** **06/12/2020 18:21:00**

Space After: 6 pt

Page 8: Deleted **Marco Petri** **06/12/2020 18:20:00**

Chapter 1 description

Page 8: Inserted **Marco Petri** **06/12/2020 18:20:00**

: this chapter introduces the problem and the system required for the realization in remarkably high terms. The chapter includes a high-level description of the stakeholder's goals and the description of the environment where the S2B is going to be realized. The chapter includes also information needed to read and understand the whole document.

Page 8: Formatted **Marco Petri** **06/12/2020 18:22:00**

English (United Kingdom)

Page 8: Deleted **Marco Petri** **06/12/2020 18:20:00**

Page 8: Formatted **Marco Petri** **06/12/2020 18:22:00**

Keyword, Italian (Italy)

Page 8: Deleted **Marco Petri** **06/12/2020 18:20:00**

Chapter 1 description

Page 8: Inserted **Marco Petri** **06/12/2020 18:20:00**

: this chapter describes extensively the system and is completely devoted to the system's description in high-level terms. The system is described under different views: scenarios of utilization of the system, definition and characterization of actors, functions to be realized in the system and the assumptions and constraints to the system.

Page 8: Deleted **Marco Petri** **06/12/2020 18:21:00**

Page 8: Formatted **Marco Petri** **06/12/2020 18:22:00**

Keyword, Italian (Italy)

Page 8: Deleted	Marco Petri	06/12/2020 18:20:00

Chapter 1 description

Page 8: Inserted	Marco Petri	06/12/2020 18:20:00

: this chapter treats all the requirements of the system and effectuates a mapping between the goals and the requirements providing a view of the satisfaction of the goals by the functions required for the system. The chapter provides a detailed description of the system attributes and describes the constraints for the realization of the system. Moreover it contains high-level interfaces' view.

Page 8: Deleted	Marco Petri	06/12/2020 18:21:00

Page 8: Inserted	Marco Petri	06/12/2020 18:21:00

Chapter 4: this chapter includes an Alloy code used to describe some system aspects formally. The Alloy language is used to describe the system and verify if some important properties of the system are satisfied by the requirements.

Chapter 5: this chapter describes using synthetic tables the effort of the team members in the process of building the document.

Chapter 6: this chapter includes information on references not cited in the first chapter.

Page 8: Formatted	Marco Petri	06/12/2020 18:22:00

Keyword, Italian (Italy)

Page 8: Formatted	Marco Petri	06/12/2020 18:22:00

Keyword, Italian (Italy)

Page 8: Formatted	Marco Petri	06/12/2020 18:22:00

Keyword, Italian (Italy)

Page 8: Formatted	Marco Petri	06/12/2020 18:22:00

Space After: 6 pt

Page 8: Deleted	Marco Petri	06/12/2020 18:21:00

Page 8: Inserted	Marco Petri	06/12/2020 18:33:00

Page 8: Deleted	Marco Petri	06/12/2020 18:21:00

Chapter 4

Chapter 1 description

Page 9: Inserted **aa m** **08/12/2020 13:49:00**

Ticket machine and in store queue

Page 9: Formatted **aa m** **08/12/2020 13:50:00**

Font: 14 pt, Bold

Page 9: Formatted **aa m** **08/12/2020 13:50:00**

Font: 14 pt, Bold

Page 9: Formatted **aa m** **08/12/2020 14:00:00**

Normal, No bullets or numbering

Page 9: Inserted **aa m** **08/12/2020 13:52:00**

Online Queue

A young woman wants to shop at a local grocery and does not have any time preference. She logs into her account, chooses to enter a queue, selects the store, the travel mode and then confirms her queueing.

Page 9: Formatted **aa m** **08/12/2020 13:58:00**

Font: 14 pt, Bold

Page 9: Moved from page 10 (Move #1) **aa m** **08/12/2020 13:55:00**

When it's time for themher to head thereto the store, theyshe receives a notification to remindremind them-her.

Page 9: Deleted **aa m** **08/12/2020 13:55:00**

them

Page 9: Inserted **aa m** **08/12/2020 13:55:00**

her

Page 9: Deleted **aa m** **08/12/2020 13:55:00**

there

Page 9: Inserted **aa m** **08/12/2020 13:55:00**

to the store

Page 9: Deleted **aa m** **08/12/2020 13:55:00**

they

Page 9: Inserted **aa m** **08/12/2020 13:55:00**

she

Page 9: Inserted **aa m** **08/12/2020 13:55:00**

s

Page 9: Deleted	aa m	08/12/2020 13:55:00

remind

Page 9: Inserted	aa m	08/12/2020 13:55:00

remind

Page 9: Deleted	aa m	08/12/2020 13:55:00

them

Page 9: Inserted	aa m	08/12/2020 13:55:00

her

Page 9: Inserted	aa m	08/12/2020 13:55:00

She then heads to the store by car, as she specified with the travel mode. Once arrived she waits for her number and goes through the entrance control once her number appears. She is allowed in and proceeds shopping.

Booking

Page 9: Formatted	aa m	08/12/2020 13:50:00

Font: 14 pt, Bold

Page 9: Formatted	aa m	08/12/2020 14:00:00

Normal, No bullets or numbering

Page 10: Moved to page 9 (Move #1)	aa m	08/12/2020 13:55:00

When it's time for them to head there, they receive a notification to remind them.

Page 10: Inserted	aa m	08/12/2020 13:47:00

Store Manager sets store's parameters

Page 10: Formatted	aa m	08/12/2020 13:50:00

Font: 14 pt, Bold

Page 10: Inserted	aa m	08/12/2020 13:43:00

Checkpoint controller at entrance

Page 10: Formatted	aa m	08/12/2020 13:50:00

Font: 14 pt, Bold

Page 10: Inserted	aa m	08/12/2020 13:47:00

Checkpoint controller at exit

A cashier is tasked to scan the ticket of the paying customers. Every time a customer pays she asks them for the ticket and scans the QR code with her phone. She diligently handle her task for all her work hours.

Customer’s registration

Page 10: Formatted	**aa m**	**08/12/2020 14:01:00**

Font: 14 pt, Bold

Page 10: Formatted	**aa m**	**08/12/2020 13:50:00**

Font: 14 pt, Bold

Page 10: Inserted	**aa m**	**08/12/2020 13:42:00**

Booking deletion

A nurse has been infected by a virus at work. She had previously booked a visit to a store, and since she is sick, she decides not to go. She logs in into her account and deletes her booking. In this way someone else can book instead of her.

Queue exit

A man who online queued to shop at a bakery decides that he would rather shop at a supermarket. He therefore logs into his account and exits from his queue, shortening the queue for those behind him. He then queues for the supermarket.

Page 10: Formatted	**aa m**	**08/12/2020 14:01:00**

Font: 14 pt, Bold

Page 10: Formatted	**aa m**	**08/12/2020 14:00:00**

Font: 14 pt, Bold

Page 13: Formatted	**Marco Petri**	**01/12/2020 22:35:00**

English (United Kingdom)

Page 13: Formatted	**Marco Petri**	**01/12/2020 22:35:00**

English (United Kingdom)

Page 13: Formatted	**Marco Petri**	**01/12/2020 22:35:00**

English (United Kingdom)

Page 13: Formatted	**Marco Petri**	**01/12/2020 22:35:00**

English (United Kingdom)

Page 13: Inserted	**Marco Petri**	**08/12/2020 18:16:00**

3

Page 13: Deleted	**Marco Petri**	**08/12/2020 18:16:00**

3

Page 13: Formatted	**Marco Petri**	**01/12/2020 22:35:00**

English (United Kingdom)

Page 19: Deleted	**asus**	**03/12/2020 06:31:00**

Access to the store by ticket (i.e. for queuing customers) is granted if and only if the customer's ticket number has been called and the next ticket number has not been called yet.

1.

Page 19: Inserted	asus	03/12/2020 06:31:00

Checkpoint controllers shall make sure that customers can enter only after proving that their ticket is valid and, in case they booked a visit, that the real size of their group does not exceed the declared size.

1.

Page 19: Inserted	asus	03/12/2020 06:30:00

There is no hidden way in which a customer can enter/exit the store.

1.

Page 19: Deleted	asus	03/12/2020 06:31:00

In particular such employee will not permit multiple groups to enter at a time and will make sure that customers who exit do not get in contact with queuing/entering customers within the area of the store.

1.

Page 19: Deleted	Marco Petri	02/12/2020 11:20:00

Each booking requires the specification of the number of people that will make the purchases within the same group.

1.

Page 19: Deleted	asus	03/12/2020 06:37:00

A checkpoint controller will scan the QR codes of every customer in order to prove the validity of their ticket/booking. Customers holding an invalid ticket will be prevented from entering the store.

1.

Page 19: Inserted	asus	03/12/2020 06:40:00

Customers respect all safety norms while waiting outside, and all checkpoint controllers will make sure to monitor that. E.g. in case a customer is not respecting social distance or is not wearing a mask, a checkpoint controller will take care of the matter.

1.

Page 19: Inserted	asus	03/12/2020 06:33:00

Customers are not permitted to exit unless such event is registered.

1.

Page 19: Deleted	asus	03/12/2020 06:33:00

This could either be done by a different employee, or even by cashiers after payment.

1.

Page 24: Inserted	asus	07/12/2020 18:36:00

$_2$: their requests for another ticket is simply overwritten.

Page 24: Deleted	asus	07/12/2020 18:36:00

. Furthermore, if a customer is in the queue for store S they cannot book a ticket for queuing for another store $S_1 \neq S$

Page 24: Deleted	asus	03/12/2020 06:46:00

estimated waiting time

Page 24: Inserted	asus	03/12/2020 06:46:00

interval

Page 24: Inserted	asus	03/12/2020 06:46:00

(

Page 24: Inserted	asus	03/12/2020 06:46:00

i, tf) representing the estimated waiting time

Page 24: Deleted	asus	03/12/2020 06:46:00

time is such that the difference between t and the real time T should not exceed 5 minutes

Page 24: Inserted	asus	03/12/2020 06:46:00

interval is calculated as a 95% confidence interval for the real waiting time T.

Page 25: Deleted	Marco Petri	08/12/2020 14:49:00

Finally, the user must input how many people are in their group.

Page 25: Inserted	asus	03/12/2020 06:49:00

Finally, the user must input how many people are in their group.

Page 25: Deleted	asus	03/12/2020 06:49:00

After booking a visit,

Page 25: Inserted	asus	03/12/2020 06:49:00

Then

Page 26: Inserted	Marco Petri	02/12/2020 11:20:00

R24	**Each booking requires the specification of the number of people that will make the purchases within**

the same group

Page 26: Inserted	**asus**	**07/12/2020 18:31:00**

R25	Customers are allowed to delete a booking at most 30 minutes before the expected time of the booking. A customer can also exit their queue at any time if they wish.

Page 26: Formatted Table	**Marco Petri**	**02/12/2020 11:36:00**

Formatted Table

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G10

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G9

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:42:00**

G12

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:42:00**

G11

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G10

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G9

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:42:00**

G12

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:42:00**

G11

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G6

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G5

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G8

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G7

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:42:00**

G11

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:42:00**

G10

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G7

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G6

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G8

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G7

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G7

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G6

Page 26: Deleted	**Marco Petri**	**06/12/2020 18:41:00**

G9

Page 26: Inserted	**Marco Petri**	**06/12/2020 18:41:00**

G8

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G7

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G6

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G9

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G8

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G10

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G9

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G6

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G5

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G6

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G5

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G8

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G7

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G7

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G6

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G13

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G12

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G13

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G12

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G7

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G6

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G14

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G13

Page 26: Deleted	Marco Petri	06/12/2020 18:42:00

G12

Page 26: Inserted	Marco Petri	06/12/2020 18:42:00

G11

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G14

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G13

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G9

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G8

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G10

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G9

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G9

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G8

Page 26: Deleted	Marco Petri	06/12/2020 18:41:00

G10

Page 26: Inserted	Marco Petri	06/12/2020 18:41:00

G9

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G13

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G12

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G13

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G12

Page 26: Deleted	Marco Petri	06/12/2020 18:42:00

G12

Page 26: Inserted	Marco Petri	06/12/2020 18:42:00

G11

Page 26: Deleted	Marco Petri	06/12/2020 18:43:00

G14

Page 26: Inserted	Marco Petri	06/12/2020 18:43:00

G13

Page 27: Deleted	Marco Petri	06/12/2020 18:41:00

G9

Page 27: Inserted	Marco Petri	06/12/2020 18:41:00

G8

Page 27: Inserted	Marco Petri	02/12/2020 11:20:00

R24	**G2**
R25	G3, G5

The previous mapping states where requirements goes, this is helpful in identifying the purpose of a requirement in the project. However, to spot the importance of a goal and to identify how the goal can be guaranteed is important to apply the inverse mapping, the mapping from goals to requirements:

Goal	Requirements	Domain assumptions
G1	R1, R3, R4, R5, R6	D2
G2	R1, R3, R4, R13, R14, R15, R24	D2
G3	R1, R4, R25	-
G4	R2, R6, R7, R8, R9, R12, R13, R16, R19, R21	D3, D4, D5
G5	R5, R9, R10, R25	D2
G6	R6, R7, R8, R11, R13	D9
G7	R5, R6, R10	D1, D2
G8	R7, R8, R17, R18, R23	D3, D11
G9	R1, R3, R8, R17, R18	D10
G10	R5	D12
G11	R2, R3, R15, R21	-
G12	R11, R12, R19, R20	D1, D5, D6, D7
G13	R14, R16, R22	-

Page 27: Formatted	Marco Petri	02/12/2020 11:36:00

Space Before: 12 pt, After: 12 pt, Line spacing: Multiple 1.07 li

Page 27: Formatted Table	Marco Petri	02/12/2020 11:36:00

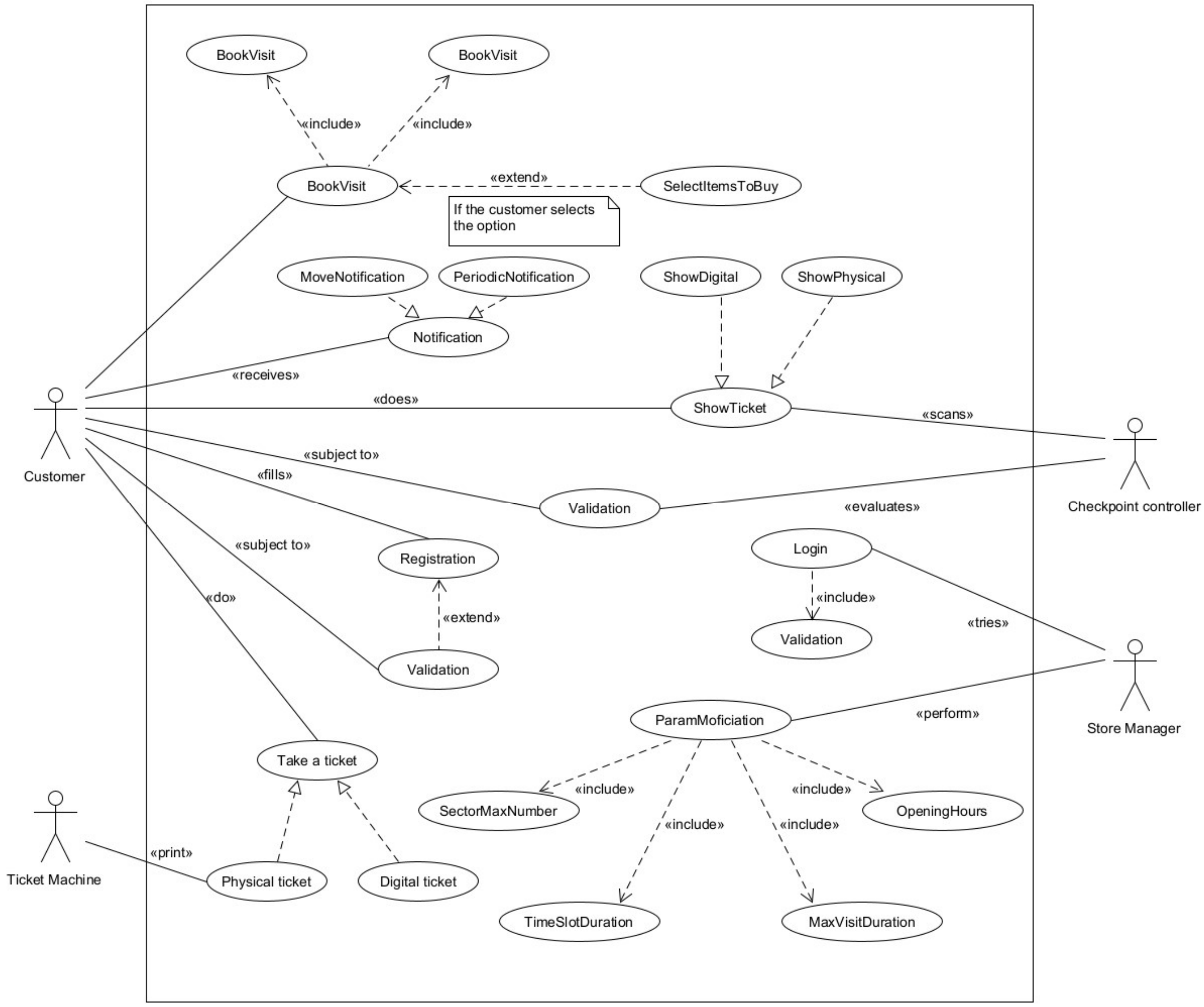
Formatted Table

Page 27: Formatted	Marco Petri	02/12/2020 11:36:00

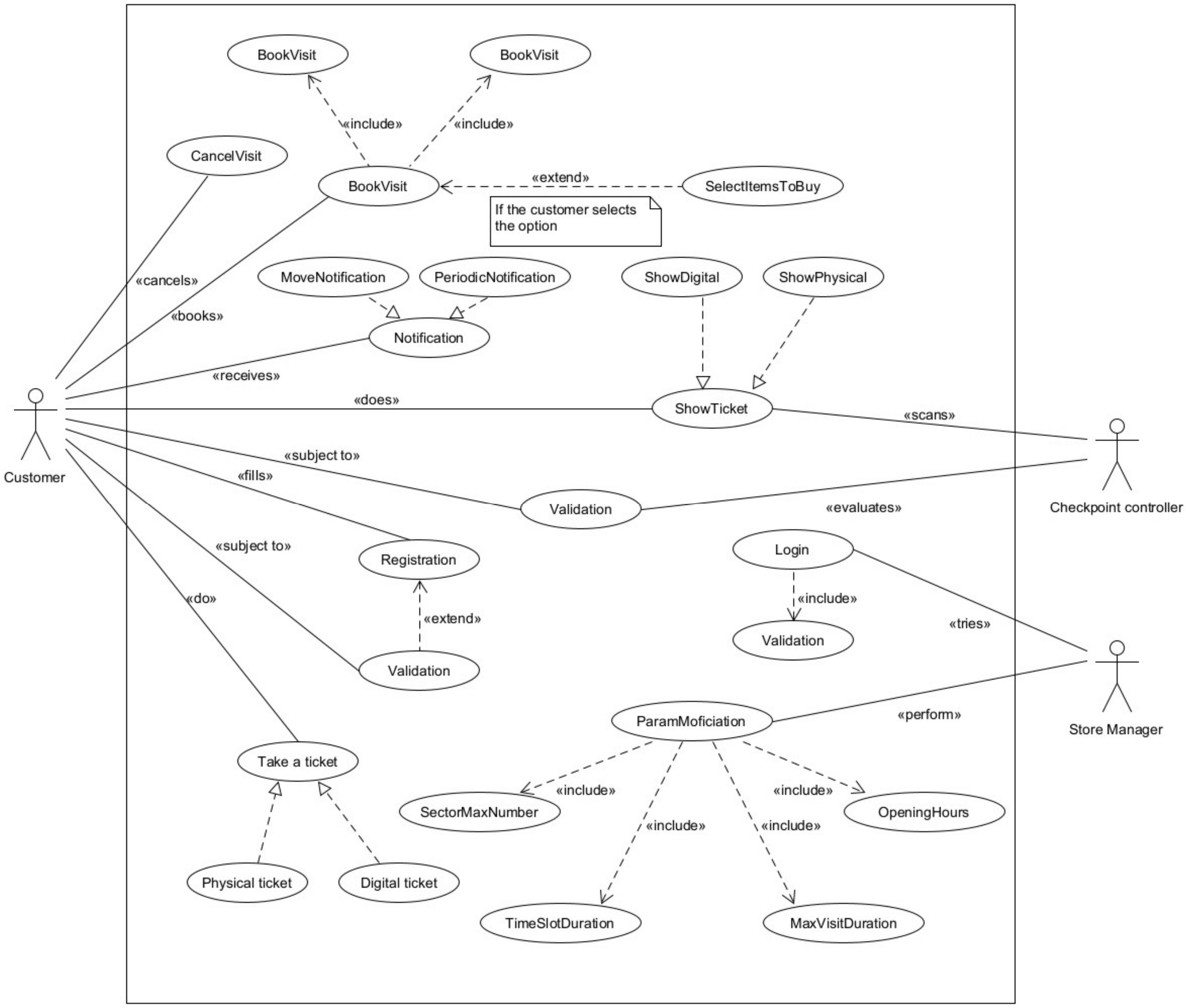
Space Before: 12 pt, After: 0 pt, Line spacing: Multiple 1.07 li

Page 28: Deleted	Marco Petri	02/12/2020 11:26:00

Page 28: Deleted	Marco Petri	02/12/2020 11:19:00



Page 28: Inserted	Marco Petri	08/12/2020 11:12:00



Page 29: Inserted	asus	03/12/2020 06:50:00

, maximum people in one group for a booking

1.

Page 30: Formatted Table	asus	07/12/2020 18:39:00

Formatted Table

Page 31: Inserted	asus	03/12/2020 06:43:00

1.

4.a If the customer already has a ticket, it is simply replaced by the new one

Page 31: Formatted	Unknown	03/12/2020 06:43:00

Indent: Left: 1.27 cm, No bullets or numbering

Page 31: Deleted	asus	03/12/2020 06:43:00

Logged Customer already has a valid ticket. The operation is deleted and an error message to remind the user that they are already queuing is displayed.

Page 32: Inserted	asus	03/12/2020 06:50:00

1.

Customer inserts the number of people that will visit the store together.

1.

Page 32: Deleted	Marco Petri	08/12/2020 11:13:00

Page 32: Inserted	asus	07/12/2020 18:41:00

Page 32: Formatted	Marco Petri	08/12/2020 11:13:00

Heading 4

Page 32: Inserted	Marco Petri	08/12/2020 11:13:00

Booking deletion

1.1.1.1

Page 32: Inserted	asus	07/12/2020 18:41:00

1.1.1.1

Use case	Delete Booking
Actor	Logged Customer
Entry condition	Customer has booked a visit and the current time is at least 30 minutes before than the visit time
Basic Flow	1. Logged Customer selects the booking. 2. Logged Customer clicks the "delete visit" button.
Exit condition	Logged Customer has successfully deleted the booking and cannot enter the store at the time of the previous booking.
Exception	

Page 32: Formatted	Unknown	07/12/2020 18:45:00

Numbered + Level: 1 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0.63 cm + Indent at: 1.27 cm

Page 32: Inserted	Marco Petri	08/12/2020 11:13:00

Queue exit

1.1.1.1

Page 32: Formatted	Marco Petri	08/12/2020 11:13:00

Heading 4

Page 32: Inserted	asus	07/12/2020 18:46:00

1.1.1.1

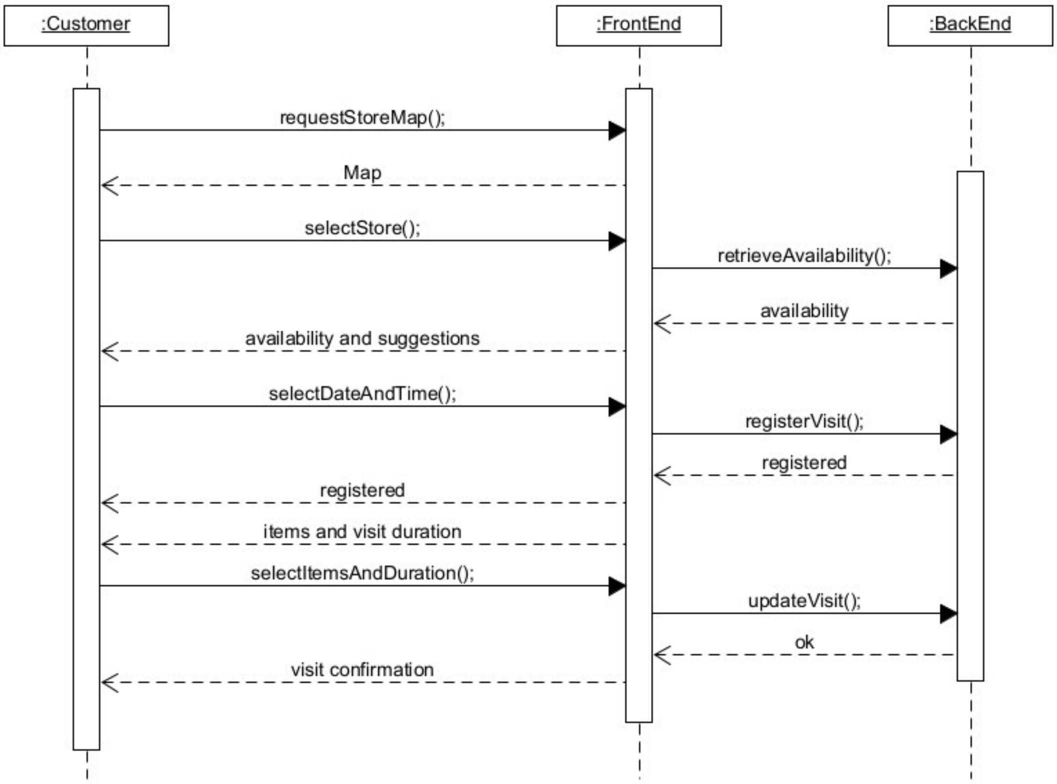
Use case	Exit Queue
Actor	Logged Customer
Entry condition	Customer is in a queue for a store and its ticket is digital
Basic Flow	1. Logged Customer enters the queue info screen section. Logged Customer clicks the "exit queue" button. 2.
Exit condition	Logged Customer is no longer in the queue. Their ticket is deleted and thus entrance is no longer permitted.
Exception	

Page 32: Formatted	Unknown	08/12/2020 11:13:00

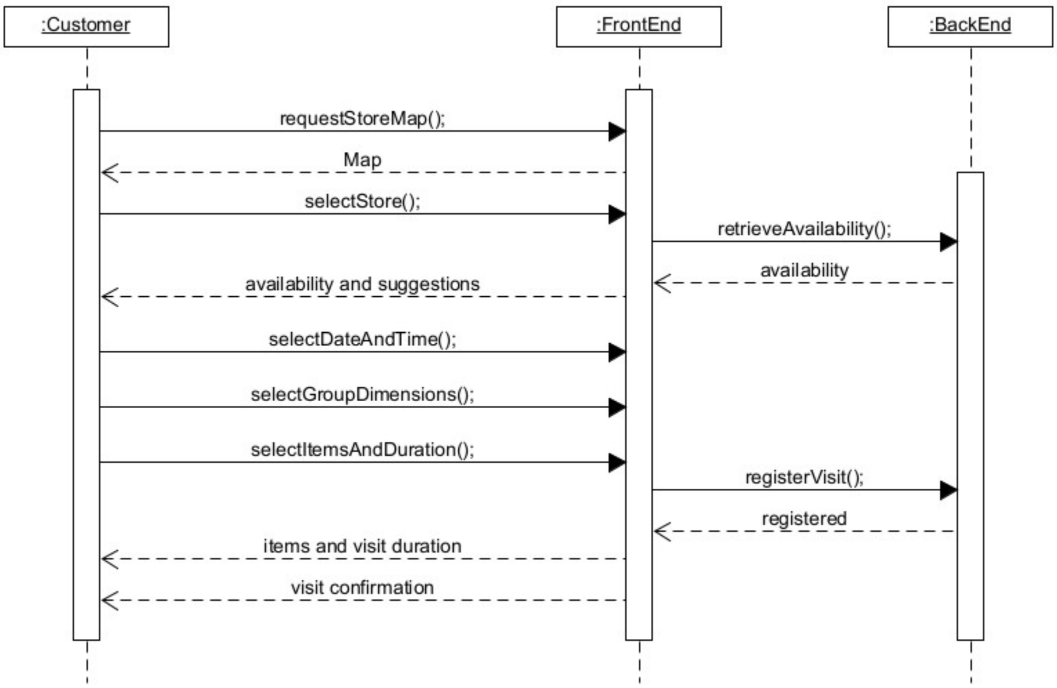
Numbered + Level: 1 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0.63 cm + Indent at: 1.27 cm

Page 32: Deleted	Marco Petri	08/12/2020 11:13:00

Page 36: Deleted	Marco Petri	02/12/2020 11:18:00

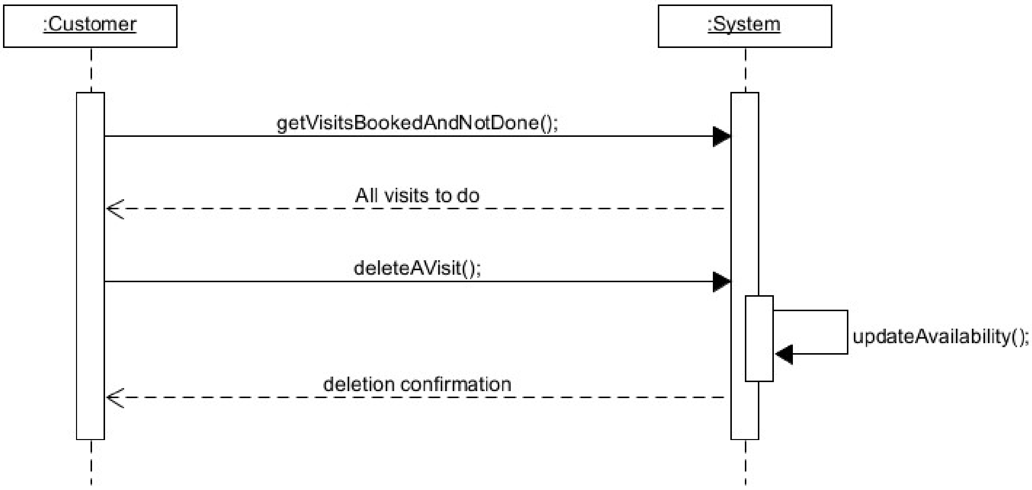


Page 36: Inserted **Marco Petri** **02/12/2020 11:18:00**



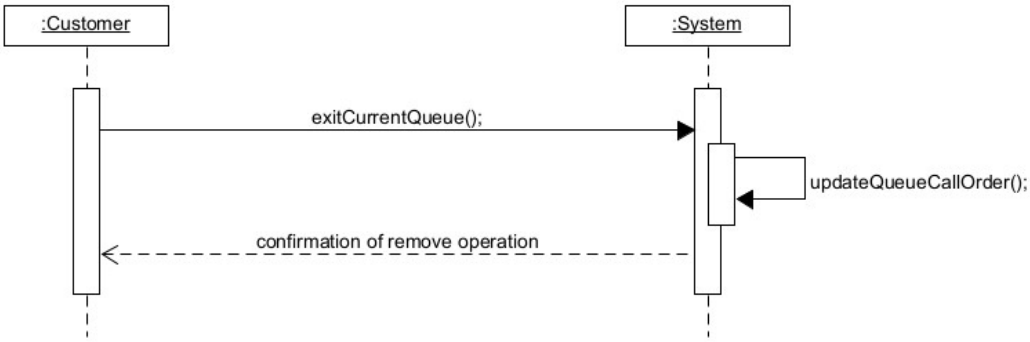
0.0.0.0 Booking deletion

Booking deletion sees customer as external actor which interacts only with the system digitally.



0.0.0.0 Queue exit

Queue exit operation is possible for all the people being in a queue online with a digital ticket. Customer is the only external actor present in this sequence derived by the use case.



Page 40: Commented [MP2]	**Marco Petri**	**28/11/2020 16:10:00**

Revisionare

Page 42: Inserted	**Marco Petri**	**02/12/2020 11:13:00**

Alloy source code

1.1

Page 42: Formatted	**Marco Petri**	**02/12/2020 11:13:00**

Heading 2

Page 42: Inserted	aa m	08/12/2020 17:34:00

```
sig Store {
      storeManagers: some StoreManager,
      storeControllers: some CheckpointController,
      storeTicketMachines: some TicketMachine,
      storeProducts: some Item,
      //customers currently inside the store
      storeCustomersInStore: set Customer,
      //customers visits at the store
      //a visit is counted when the customer enters the store
      storeCustomersVisits: set Visit,
      //current queue for the store
      storeQueue: one Queue,
      //current bookings for the store
      storeBookings: set Booking,
      //tickets that have been used to enter the store, but not necessarely
to exit
      storeUsedTickets: set Ticket,
      //tickets that have not been used to enter the store, and are no more
valid
      storeNotUsedInvalidTickets: set Ticket
}{
      #storeControllers>0
      #storeTicketMachines>0
}


sig Visit{
      visitTicket: one Ticket,
      visitDate: one Date,
}

abstract sig Person {}

sig CheckpointController extends Person {
      controllerControlsEntrance: set Ticket,
      //rejected tickets
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      controllerControlsExit: set Ticket,
}

sig StoreManager extends Person {}

sig Customer extends Person {}

sig TicketMachine {
      machinePrintedTickets: set QueueTicket
}

sig Category {}

sig Item {
      itemCategory: one Category
```

```
}

abstract sig Ticket {
	ticketOwner: one Customer,
	ticketQRCode: one QRCode,
}

sig Booking {
	bookingTicket: one BookingTicket,
	bookingDate: one Date,
	bookingItems: set Item,
	bookingCategories: set Category
}

sig BookingTicket extends Ticket {}

sig QueueTicket extends Ticket {}

sig Queue {
	queueNumbers: set QueueNumber,
	queueFirstNumberInLine: lone QueueNumber,
}



sig QueueNumber {
	queueNumberTicket: one QueueTicket,
	queueNumberNext: lone QueueNumber,
}

sig QRCode {}{
	//every TicketNumber corresponds to exactly one QueueTicket
	one q: Ticket | this = q.ticketQRCode
}

sig Date {}

//FACTS

fact  customersFacts{
	//customers in store have been checked in but not checked out
	all c: Customer | all s: Store | one t: Ticket |
	c in s.storeCustomersInStore implies
	(c = t.ticketOwner and
	t in s.storeControllers.controllerControlsEntrance and
	t not in s.storeControllers.controllerControlsExit)
	//customers have max one queueTicket  Fixare!!!!!!!!!!!!!!!!!!!
	//tanti booking non si sovrappongo
	all c: Customer | lone t: Ticket | c = t.ticketOwner
}

fact queueFacts{//metti numbers!!!!!!!!!!!!!!!!
	//a first ticket is not the next of any ticket
```

```
	all q: Queue | no qt: QueueNumber | qt.queueNumberNext = q.queueFirst-
NumberInLine
	//if there are at least than 1 tickets there must be a first ticket
	all q: Queue | #q.queueNumbers > 0 implies #q.queueFirstNumberInLine =
1
	//if there is a firstTicket it must belong to the queue tickets
	all q: Queue | q.queueFirstNumberInLine in q.queueNumbers
}

fact queueTicketsFacts{
	//tickets' next not reflexive
	no qt: QueueNumber | qt.queueNumberNext = qt
	//tickets' next not cyclic
	no qt: QueueNumber | qt in qt.^queueNumberNext
	//tickets are connected to the others in the same queue
	all qt: QueueNumber | one q: Queue | qt in q.queueFirstNumber-
InLine.*queueNumberNext
	and qt in q.queueNumbers
	//no shared next tickets
	all disj qt1,qt2,qt3: QueueNumber | qt1.queueNumberNext = qt3 implies
qt2.queueNumberNext != qt3
	//queueTickets are assigned to one queue
	//all qt: QueueTicket | one q: Queue | qt in q.queueTickets
}


fact storeEntitiesAssignedToAStore{
	all sm: StoreManager | some s: Store | sm in s.storeManagers
	all cp: CheckpointController | one s: Store | cp in s.storeControllers
	all i: Item | some s: Store | i in s.storeProducts
	all c: Category | some i: Item | c = i.itemCategory
	all q: Queue | one s: Store | q in s.storeQueue
	all b: Booking | one s: Store | b in s.storeBookings
	all tm: TicketMachine | one s: Store | tm in s.storeTicketMachines
}


fact ticketsFacts{
	//tickets are assigned a store:
	all t: Ticket | one s: Store | t in (s.storeUsedTickets + s.store-
NotUsedInvalidTickets +
	s.storeBookings.bookingTicket + s.storeQueue.queueNumbers.queueNumber-
Ticket)

	//tickets can either be:
	all t: Ticket | all s: Store |
	//s.storeUsedTickets&(s.storeNotUsedInvalidTickets+
	//used                    FIXARE col + e magari migliorare
	(t in s.storeUsedTickets implies (t not in s.storeNotUsedInvalidTick-
ets and
	t not in s.storeQueue.queueNumbers.queueNumberTicket and t not in
s.storeBookings.bookingTicket)) and
	//not used and invalid
```

```
      (t in s.storeNotUsedInvalidTickets implies (t not in s.storeUsedTick-
ets and
      t not in s.storeQueue.queueNumbers.queueNumberTicket and t not in
s.storeBookings.bookingTicket)) and
      //valid for booking
      (t in s.storeBookings.bookingTicket implies (t not in s.store-
NotUsedInvalidTickets and
      t not in s.storeQueue.queueNumbers.queueNumberTicket and t not in
s.storeUsedTickets)) and
      //valid for queueing
      (t in s.storeQueue.queueNumbers.queueNumberTicket implies (t not in
s.storeNotUsedInvalidTickets and
      t not in s.storeUsedTickets and t not in s.storeBookings.booking-
Ticket))

      //tickets have an owner
      all t: Ticket | one c: Customer | t.ticketOwner = c
      //if tickets are printed, they are printed by a machine of the store
they are for
      all t: Ticket | all s: Store | t in s.storeTicketMachines.machine-
PrintedTickets
      implies (t in s.storeBookings.bookingTicket or t in s.store-
Queue.queueNumbers.queueNumberTicket
      or t in s.storeNotUsedInvalidTickets or t in s.storeUsedTickets)
      //tickets can be controlled only once at the entrance
      all t: Ticket | lone c: CheckpointController | t in c.controllerCon-
trolsEntrance
      //tickets can be controlled only once at the exit
      all t: Ticket | lone c: CheckpointController | t in c.controllerCon-
trolsExit
      //tickets can only be printed once
      all t: Ticket | lone tm: TicketMachine | t in tm.machinePrintedTickets
      //all tickets checked at the exit have been checked at the entrance
      all t: Ticket | all s: Store |
      t in s.storeControllers.controllerControlsExit implies t in s.store-
Controllers.controllerControlsEntrance
      //all used tickets have been checked at entrance
      all t: Ticket | all s: Store |
      t in s.storeUsedTickets iff t in s.storeControllers.controllerCon-
trolsEntrance
}

fact visitsFacts{
      //all visits are assigned to one store
      all v: Visit |  one s: Store |  v in s.storeCustomersVisits
      //a visit was made if and only if its ticket was checked in by the
controller
      all s: Store | all v: Visit |
      v.visitTicket in s.storeControllers.controllerControlsEntrance iff v
in s.storeCustomersVisits
      //every time a controller checks in a ticket only one visit is counted
      all s: Store | all t: Ticket | one v: Visit |
      t in s.storeControllers.controllerControlsEntrance implies
```

```
      (v in s.storeCustomersVisits and v.visitTicket = t)
}

 //ASSERTIONS

//a ticket can only be used for max a single visit
assert ticketMaxOneVisit{
      all t: Ticket | lone v: Visit |
      t = v.visitTicket
}
check ticketMaxOneVisit for 6

//all used tickets of a store have been used for visiting that store
assert usedTicketForVisitOfSameStore{
      all s: Store | all v: Visit |
      v in s.storeCustomersVisits implies v.visitTicket in s.storeUsedTick-
ets
}
check usedTicketForVisitOfSameStore for 6

//all customers in queue cannot enter before their ticket is called
assert customersRespectQueue{
      all s: Store | all qt: QueueTicket | all c: Customer |//RIGUARDARE
      (c = qt.ticketOwner and qt in s.storeQueue.queueFirstNumber-
InLine.*queueNumberNext.queueNumberTicket)
      implies c not in s.storeCustomersInStore
}
check customersRespectQueue for 6

//no customer has tickets for more than one queue
assert noCustomerInTwoQueues {
      all disj q1,q2: Queue | all c: Customer | c in q1.queueNumbers.queue-
NumberTicket.ticketOwner
      implies c not in q2.queueNumbers.queueNumberTicket.ticketOwner
}
check noCustomerInTwoQueues for 6

//no customer has more than one tickets for the same queue
assert noCustomerTwiceInQueue{
      all q: Queue | no disj t1, t2: QueueTicket | t1 in q.queue-
Numbers.queueNumberTicket
      and t2 in q.queueNumbers.queueNumberTicket
      and t1.ticketOwner = t2.ticketOwner
}
check noCustomerTwiceInQueue for 6



// PREDICATES

//visite non possono avere stesso date and time

//
```

```
//customers can use the ticket machine to visit the store they want
pred customersCanUseTicketMachineToVisit{
	#Store>0
	all s: Store | some v: Visit |
	v in s.storeCustomersVisits and
	v.visitTicket in s.storeTicketMachines.machinePrintedTickets
}
run customersCanUseTicketMachineToVisit for 6

//customers can use online queue to visit the store they want
pred customersCanUseOnlineQueueToVisit{
	#Store>0
	all s: Store | some v: Visit | some qt: QueueTicket |
	v in s.storeCustomersVisits and
	v.visitTicket = qt and
	v.visitTicket not in s.storeTicketMachines.machinePrintedTickets
}
run customersCanUseOnlineQueueToVisit for 6

//customers can use online booking to visit the store they want
pred customersCanUseOnlineBookingToVisit{
	#Store>0
	all s: Store | some v: Visit | some bt: BookingTicket |
	v in s.storeCustomersVisits and
	v.visitTicket = bt
}
run customersCanUseOnlineBookingToVisit for 6

pred show {}

/*run {
	#Store = 2
	#Date = 4
	#Item = 2
	#Category = 1
	#Queue = 2
	#Booking = 1
	#TicketMachine = 3
	#Customer = 3
	#QueueTicket = 2

} for 5*/


run show

//run show for exactly 2 Store, 4 Person, 2 TicketMachine, 2 Category,
//2 Item, 15 Ticket, 1 Booking, 5 Queue, 5 TicketNumber, 15 QRCode, 1 Date

//run show for 11 but 3 Date, 3 Item, 3 Customer, 4 CheckpointController, 3
Booking
```

Page 48: Deleted **aa m** **08/12/2020 17:34:00**

```
abstract sig Person {}
abstract sig Ticket {
	ticketQRCode: one QRCode,
	ticketOwner: one Customer,
	ticketStoreRef: one Store,
}

/*	Here there are the humans entities composing the system: customers,
	store managers and checkpoint controllers. They have been defined
	using only what characterize them. */
sig Customer extends Person {
	customerTickets: set Ticket,
	customerBookings: set Booking,
	customerVisits: set Store
} {
	// if a customer has visited at least one store, it must have at least one ticket
	#customerTickets >= #customerVisits
	// all owners of tickets are this customer
}
sig CheckpointController extends Person {
	controllerStore: one Store,
	controllerControls: set Ticket
}
sig StoreManager extends Person {
	managerStores: some Store
}

/*	In this part there are the definitions of physical objects of the
	system and some abstract concepts related to the stores. Objects
	contains the stores, tickets, reservations (done by customers) and
	the abstract concept of the visit. Which is strictly related
	to the presence of a ticket and a customer. */
sig Store {
	storeManager: some StoreManager,
	storeControllers: some CheckpointController,
	storeTicketMachines: some TicketMachine,
	storeTickets: set Ticket,
	storeProducts: some Item
} {
	// tickets printed by the ticket machines must be a subset of all tickets for the store
	storeTicketMachines.machinePrintedTickets in storeTickets
	storeControllers.controllerControls in storeTickets
}
sig TicketMachine {
	machineStore: one Store,
	machinePrintedTickets: set QueueTicket
}
sig Category {}
sig Item {
	itemCategory: one Category
```

```
}

/*	Ticket entities and entities related to tickets. */
sig Booking {
	bookingOwner: one Customer,
	bookingTicket: one BookingTicket,
	bookingDate: one Date,
	bookingItems: set Item,
	bookingPeople: some Customer,
	bookingCategories: set Category
}
sig Queue {
	queueStore: one Store,
	queueTickets: set QueueTicket,
	queueNextCall: one QueueTicket
}
sig TicketNumber {}
sig QRCode {}
sig QueueTicket extends Ticket {
	queueTicketNumber: one TicketNumber,
	queueTicketQueue: one Queue,
	queueTicketPrevious: lone QueueTicket,
	queueTicketNext: lone QueueTicket
}
sig BookingTicket extends Ticket {
	bookTicketBooking: one Booking
}

/*	In Alloy the base type int has a very restricted domain and can't
	be used like in programming languages. As Daniel Jackson says, with
	Alloy and FOL there is no need to use integers to describe the
	system. Reference: Software Abstractions - DJ: pp. 134-135 */
sig Date {}
```

Page 48: Inserted Marco Petri 02/12/2020 11:13:00

```
abstract sig Person {}
abstract sig Ticket {
	ticketQRCode: one QRCode,
	ticketOwner: one Customer,
	ticketStoreRef: one Store,
}

/*	Here there are the humans entities composing the system: customers,
	store managers and checkpoint controllers. They have been defined
	using only what characterize them. */
sig Customer extends Person {
	customerTickets: set Ticket,
	customerBookings: set Booking,
	customerVisits: set Store
} {
	// if a customer has visited at least one store, it must have at least one ticket
	#customerTickets >= #customerVisits
```

```
	// all owners of tickets are this customer
}
sig CheckpointController extends Person {
	controllerStore: one Store,
	controllerControls: set Ticket
}
sig StoreManager extends Person {
	managerStores: some Store
}

/*	In this part there are the definitions of physical objects of the
	system and some abstract concepts related to the stores. Objects
	contains the stores, tickets, reservations (done by customers) and
	the abstract concept of the visit. Which is strictly related
	to the presence of a ticket and a customer. */
sig Store {
	storeManager: some StoreManager,
	storeControllers: some CheckpointController,
	storeTicketMachines: some TicketMachine,
	storeTickets: set Ticket,
	storeProducts: some Item
} {
	// tickets printed by the ticket machines must be a subset of all tickets for the store
	storeTicketMachines.machinePrintedTickets in storeTickets
	storeControllers.controllerControls in storeTickets
}
sig TicketMachine {
	machineStore: one Store,
	machinePrintedTickets: set QueueTicket
}
sig Category {}
sig Item {
	itemCategory: one Category
}

/*	Ticket entities and entities related to tickets. */
sig Booking {
	bookingOwner: one Customer,
	bookingTicket: one BookingTicket,
	bookingDate: one Date,
	bookingItems: set Item,
	bookingPeople: some Customer,
	bookingCategories: set Category
}
sig Queue {
	queueStore: one Store,
	queueTickets: set QueueTicket,
	queueNextCall: one QueueTicket
}
sig TicketNumber {}
sig QRCode {}
sig QueueTicket extends Ticket {
```

```
        queueTicketNumber: one TicketNumber,
        queueTicketQueue: one Queue,
        queueTicketPrevious: lone QueueTicket,
        queueTicketNext: lone QueueTicket
}
sig BookingTicket extends Ticket {
        bookTicketBooking: one Booking
}

/*      In Alloy the base type int has a very restricted domain and can't
        be used like in programming languages. As Daniel Jackson says, with
        Alloy and FOL there is no need to use integers to describe the
        system. Reference: Software Abstractions - DJ: pp. 134-135 */
sig Date {}
```

Page 48: Deleted	Marco Petri	02/12/2020 11:13:00

```
abstract sig Person {}

/*      Here there are the humans entities composing the system: customers,
        store managers and checkpoint controllers. They have been defined
        using only what characterize them. */
sig Customer extends Person {
        customerTickets: set Ticket,
        customerBookings: set Booking,
        customerVisits: set Store
} {
        // if a customer has visited at least one store, it must have at least one ticket
        #customerTickets >= #customerVisits
}
sig CheckpointController extends Person {
        controllerStore: one Store,
        controllerControls: set Ticket
}
sig StoreManager extends Person {
        managerStores: some Store
}

/*      In this part there are the definitions of physical objects of the
        system and some abstract concepts related to the stores. Objects
        contains the stores, tickets, reservations (done by customers) and
        the abstract concept of the visit. Which is strictly related
        to the presence of a ticket and a customer. */
sig Store {
        storeAllowedTimeToGetIn: some Time,
        storeAllowedTimeToGetOut: some Time,
        storeManager: some StoreManager,
        storeControllers: some CheckpointController,
        storeTicketMachines: some TicketMachine,
        storeTickets: set Ticket,
        storeProducts: some Item
} {
        // there must be at least one time at which you can get in and get out by the store
```

```
		#(storeAllowedTimeToGetIn&storeAllowedTimeToGetOut) >= 1
		// tickets printed by the ticket machines must be a subset of all tickets for the store
		storeTicketMachines.machinePrintedTickets in storeTickets
		storeControllers.controllerControls in storeTickets
}
sig TicketMachine {
		machineStore: one Store,
		machinePrintedTickets: set Ticket
}
sig Category {}
sig Item {
		itemCategory: one Category
}
sig Booking {
		bookingOwner: one Customer,
		bookingTicket: one Ticket,
		bookingDate: one Date,
		bookingEndTime: one Time,
		bookingItems: set Item,
		bookingCategories: set Category
}
sig TicketNumber {}
sig Ticket {
		ticketNumber: one TicketNumber,
		ticketOwner: one Customer,
		ticketStoreRef: one Store,
		ticketBooking: lone Booking,
		ticketQueue: lone Queue,
		ticketDate: one Date
} {
		// a ticket is for a queue or for a booking, not both
		#ticketBooking = 1 => #ticketQueue = 0
		#ticketQueue = 1 => #ticketBooking = 0
}
sig Queue {
		queueStore: one Store,
		queueTickets: set Ticket,
		queueNextCall: one Ticket
}

/*		In Alloy the base type int has a very restricted domain and can't
		be used like in programming languages. As Daniel Jackson says, with
		Alloy and FOL there is no need to use integers to describe the
		system. Reference: Software Abstractions - DJ: pp. 134-135 */
sig Year {}
sig Month {}
sig Day {}
sig Time {}
sig Date {
		year: one Year,
		month: one Month,
		day: one Day,
```

```
        time: one Time
}
```

Header and footer changes

Page 1: Inserted	Marco Petri	08/12/2020 18:16:00

Purpose

Page 1: Deleted	Marco Petri	08/12/2020 18:16:00

Alloy source code

Page 1: Inserted	Marco Petri	08/12/2020 18:16:00

Purpose

Page 1: Deleted	Marco Petri	08/12/2020 18:16:00

Alloy source code

Text Box changes

Header and footer text box changes

Footnote changes

Endnote changes