1. Introduction

a. Purpose

This document is the Requirement Analysis and Specification Document of the project Travlendar+, a calendar-based application that schedules users’ appointments and support them in arranging travels.

The present is meant to be a guide-line for the implementation of the application, since it contains a deep analysis of the requirements, the goals and the domain of the environment surrounding the system and the users.

The analysis is carried out through the use of natural language, whose drawbacks and inconsistency will be amended by formalisms like UML and Alloy.

The application is meant to satisfy the following goals:

G1. Allow the user to register/login with username, password and email.

G.1.1 The user can login in the application with an existing account;

G.1.2 The user can create a new account;

G.1.3 The user can recover its lost password using the email used in the registration process.

G2. Customize the user account with details for travel planning.

G.2.1. The user can customize its home location;

G.2.2. The user can customize its breaks during the day, pointing out its lunch;

G.2.3. The user can insert its personal vehicles;

G.2.4. The user can insert its transportation passes;

G.2.5. The user can customize its preferences about transportation G.2.6. The user can insert its vehicle sharing subscriptions;

G3. Schedule user’s appointments along the days.

G.3.1. The user can insert a new appointment in its calendar.

G.3.2. The user can view its appointments along the calendar.

G.3.2. The user can delete an appointment.

G.3.4. The user can edit the details of an appointment.

G4. Notify the user of incoming appointments, with details about starting time and mean of transportation.

G5. Propose a suitable itinerary throughout the appointments’ locations, according to user’s preferences, appointment description and external information about public transportation, weather forecast.

G.5.1. The user can view the itinerary, composed by all the details about the travel among the locations of the appointments.

G.5.2. The user can edit the proposed itinerary.

G6. Give the possibility to buy a transportation ticket.

G7. Give the possibility to reserve a vehicle-sharing service.

G.7.1. Locate all the sharing-vehicles and show them on the map.

G.7.2. Let the user open the relative vehicle-sharing app to reserve a vehicle.

G8. Guide the user through all the appointments’ locations as a GPS navigator.

G.8.1. Notify real-time unexpected events, such as weather changes, car incidents and possible delays too.

G.8.2. Edit the itinerary basing it on the real current position.

b. Scope

The main goal of this service is to support people in arranging their days. It provides a calendar that can be filled by users with all their appointments, specifying time, estimated duration, location, type and possible breaks.

Once the user has inserted all this information, and according to its preferences, the system checks if it is compatible with others commitments and if a correct scheduling is feasible.

In addition to the data inserted by the user, the system will take into account many other variables (such as weather forecast, strikes, availability of private/public transportation) to compute the most efficient and suitable approach to every meeting, proposing a starting time, an estimated arrival time and the alternatives for travelling.

This application goes deep into the organization of the trip, indeed it also permits to buy a ticket for public transportation or to locate the nearest vehicle-sharing service.

By analyzing the problem the following phenomena are classified:

|  |  |  |
| --- | --- | --- |
| World phenomena | Shared phenomena | Machine phenomena |
| The user plans an appointment | The user’s position x | Most suitable path computation |
| The user starts its trip | Weather forecast changes x | Meetings consistency checking |
| Meeting delay | The user creates an account x | Checking whether a ticket must be purchased |
| The user pays for vehicle sharing | The user edits its settings x | Appointment a = new Appointment (loc, start, end, type, breaks) |
| Weather conditions changing during the trip | The user creates a new meeting x |  |
| The user reaches its destination | The user edits a meeting x |  |
| Unexpected event occurs during the trip | The user cancels a meeting x |  |
| The route is trafficked. | The system proposes an itinerary x |  |
|  | The user edits the system’s itinerary x |  |
|  | The user accepts the system’s itinerary x |  |
|  | The user purchases public transportation ticket x |  |
|  | The app displays a warning for a meeting located in an unreachable place in the allotted time, or if overlaps with other appointments.  x |  |
|  | Public transportation schedules x |  |
|  | The system notifies the user if an appointment is going to be held x |  |
|  | The app detects and displays all the vehicle-sharing services nearby x |  |
|  | GPS detects a position different from the foreseen one x |  |
|  | Google Maps detects traffic conditions in your trip x |  |

c. Definitions, Acronyms, Abbreviations

d. Revision history

e. Reference Documents

f. Document Structure

2. OVERALL DESCRIPTION

1. Product perspective (uml e statecharts su cosa? come devo approfondire gli shared phenomena?) (inserire tra i paragrafi mookups e gli altri diagrammi)

This system is expected to support users in scheduling their meetings.

//The user creates an account

The first time the user opens the app it must register, providing all the necessary details that will be processed by the system to accomplish his tasks.

The necessary information include username (unique for each user), password and all the other details useful to arrange the appointments.

The next times the user will be able to access using its credentials or to recover its password using its email address.

//The user edits its settings

If some other preferences fits the user’s needs, it user will always be allowed to edit them (but for its username).

//The user creates a new meeting

Once the user has initialized the application, it is allowed to create appointments and to fill its calendar. For each new appointment the user is asked to fill a form in order to give a description of the event, that will also be processed by the application to efficiently plan the daily scheduling.

//The app displays a warning for an appointment located in an unreachable place in the allotted time, or if overlaps with other appointments.

At the end of the creation of each appointment the system checks the consistency of the daily schedule to verify if there is enough free time to insert it and, if an inconsistency is found, the application will show a warning to the user.

In order to perform this consistency checking, the system scans all the already inserted appointments and verifies whether they overlap and/or there is enough time between two adjacent events to move from one to another (also considering the lunch and the further breaks).

If the going-to-be-inserted appointment does not overlaps with the others, but it is unreachable in the allotted time, the user has still to option to schedule it: the system will take care of letting the user reach the new appointment’s location as soon as possible.

//The system proposes an itinerary

//Weather forecast changes

//Public transportation schedules

Once the daily schedule is filled, the system can compute an itinerary throughout the various appointments: the application will propose travels between consecutive appointments, basing its analysis on the location where the appointments will be held, also considering fundamentals factors such as the most suitable mean of transportation, weather forecast and all the user’s preferences.

Furthermore, the application will take into account also the type of appointment, making yourself sure to be in time for a job interview and tolerating a little delay for a chess course.

For the events that have no starting point in their description, the default option for the itinerary is the user’s home for the first appointment of the day, or previous appointment’s location for the others.

//The user accepts the system’s itinerary.

//The user edits the system’s itinerary.

The user can both accept the system’s proposal or to modify it, changing some settings, like mean of transportation or starting time, and the system will check if the new itinerary is valid or not (if it is not, the original one is maintained).

//The user edits an appointment

//The user cancels an appointment

//The system proposes an itinerary

In every moment, the user is able to edit the already filled schedule.

After its modifications, the system will propose another itinerary.

//The system notifies the user if an appointment is going to be held

In a given day, the system will notify the user with its appointments, suggesting to start its travel at a certain time with a specific mean of transportation.

//The user’s position

//GPS detects a current position different from the foreseen one

//Weather forecast changes

//Google Maps detects traffic conditions in your route

By starting your itinerary, the application will guide you as a GPS navigator.

Actually, to better fit real-time conditions, when the application is eventually opened, a new and more accurate itinerary can be proposed, depending on the current position, possible sharing-vehicles nearby, traffic conditions and weather information.

During the travel, if an unscheduled event affects the itinerary (for instance, unexpected breaks or weather change), the system will collect the foreseen delay and will propose a feasible solution to fix the track, if there is one.

Otherwise the original track is maintained.

//The user purchases public transportation ticket

Each time a public transportation is foreseen, the application notifies that a ticket is needed, and the user has the possibility to purchase one, unless it already has a pass.

//The app detects and displays all the vehicle-sharing services nearby

On the other hand, when the travel is going to start, the application will look for possible vehicle-sharing services in the neighbours of the user’s location: if there is one (and it is convenient for the schedule), it is signaled on the map and the user, by clicking on it, can open the relative app to reserve a vehicle.

b. Product functions

G1. Allow the user to register/login with username, password and email.

R.1. If the user is already registered, the system must allow the user to authenticate himself through username and password.

R.1. When the app is opened for the first time, the system must allow the user to enter its username, password and email.

R.2. Once the user fills the required information, an auto-generated the system must send him an email to him with a link to verify the account.

R.3. Once the user clicks on the link sent by email,the system inserts the account into the database and the account is created.

G2. Customize the user account with details for travel planning.

R.0. The user must be registered and logged in.

R.1. The system must allow the user to enter its home location and to retrieve it through GPS.

R.x The system must allow the user to enter a maximum cost for trip.

R.2. The system must allow the user to enter two 30 minutes break during the day.

R.2. The system must allow the user to enter a certain period of time in which a specific mean of transportation is avoidable/preferable.

R.3. The system must allow the user to enter its transportation subscriptions.

R.x. The system must allow to enter a maximum distance coverable for each mean of transportation.

R.4. The system must allow the user to specify a time of the day devoted to have lunch.

R.5. The system must allow the user to insert all its personal vehicles.

R.6. The system must allow the user to choose an “eco-plan” for its journeys.

R.7. The system must allow the user to enter a maximum-distance coverable for a certain mean of transportation.

R.8. The system must allow the user to enter its 3 favourites mean of transportation (by foot, car, bike, motorbike, public transportation, vehicle-sharing).

G3. Schedule user’s appointments along the days.

R.0. The user must be registered and logged in.

R.9. The system must allow the user to enter the entire description of its appointments, in terms of:

* Title
* Type (the type defines the default priority of the event and some constraints on the trip).
* Date
* Starting time
* Duration
* Location
* Starting location (by default, if it is the first appointment of the day it is home, otherwise it will be the location of the previous appointment)
* Notes

R.10. After the submit of a new appointment, the system must scan all the already present appointments, checking if there is an overlap or it is impossible to reach one of them in time if the current appointment is scheduled.

R.11. If an inconsistency is found, the system must warn the user, asking him to to edit/delete the new appointment.

R.12. The system must allow the user to see all its inserted appointments.

R.13. The system must allow the user to delete each of its inserted appointments.

R.14. After the edit of an appointment, the system must re-scan all the already present appointments, checking if there is an overlap or it is impossible to reach one of them in time if the current appointment is scheduled.

R.15. The system must allow the user to edit each one of its appointments.

G4. Notify the user of incoming appointments, with short details about starting time and mean of transportation.

R.0. The user must be registered and logged in.

R.16. The system must warn the user 30 minutes before the starting time of each trip.

R.17. The system must notify the user at the starting time that it has to leave.

R.21. The system must notify a possible change in the track, due to the presence of an available shared-vehicle in the neighborhood.

G5. Propose a suitable itinerary throughout the appointments’ locations, according to user’s preferences, appointment description and external information about public transportation, weather forecast.

R.0. The user must be registered and logged in.

R.22. The system must retrieve all the necessary information about the map from an external service (such as Google Maps).

R.23. The system must contact the servers of the public transportation service of the area to retrieve the necessary information of rides.

R.24. The system must contact a weather forecast service to retrieve information about the weather in the day of the appointment.

R.25. The system must use an algorithm to produce suitable tracks, to reach the destination in time, considering all the information taken by external outsources (public transportation’s ride, weather conditions) and all the constraints given by the user’s preferences in order to minimize the duration.

R.X. The system must consider the priority of the appointments during the scheduling.

The priority is an integer number between 1 and 5. If the user does not insert a priority, the appointment gains the default value of his type.

The priority influences the scheduling in terms of assurance to be in time at the appointment’s location: higher is the priority more are the minutes in advance rather than the standard arrival time.

|  |  |
| --- | --- |
| Priority | Minutes in advance |
| 5 | 60 |
| 4 | 45 |
| 3 | 30 |
| 2 | 15 |
| 1 | 0 |

R.X. The system must consider the type of the appointments during the scheduling: [Generic appointment, business meeting, party, family appointment, dates (appointment with more than one person in the trip), office/school time, gym/workout].

|  |  |  |
| --- | --- | --- |
| Type | Default Priority | Additional constraints |
| generic appointment | 2 |  |
| business meeting | 5 |  |
| party | 1 |  |
| family appointment | 2 |  |
| dates | 4 |  |
| office/school time | 3 |  |
| gym/workout | 1 |  |

R.26. The system must show the 5 best tracks: the most ecologic, the cheapest, the shortest, the one with less walking distance and the one with minimum changes.

R.27. The system must allow the user to choose one of the track, the default option is the shortest one.

R.28. The system must save in the database the choice.

R.29. The system must take into account the priority of the appointment in calculating ETA: the higher is the priority, the greater must be the anticipation to the appointment.

G6. During a trip, when the user has to use a public transportation service, give the possibility to buy a transportation ticket.

R.0. The user must be registered and logged in.

R.29. The system must check the presence in the user’s account of a pass for that transportation service.

R.30. In absence of the pass, the system must inform the user of the possibility to buy a ticket.

R.31. If the user choose to buy the ticket, the system must perform a PayPal transaction between the user and the public transportation service.

G7. Give the possibility to reserve a vehicle-sharing service.

R.0. The user must be registered and logged in.

R.32. Every time the user open the application, the system must contact the vehicle-sharing service to retrieve the information about the location the vehicles nearby the user.

R.20. At the starting time of a trip, the system must look for shared-vehicle and, in case, verify if they are a suitable alternative to the previous track and show them on the map.

R.34. If the user selects a sharing-vehicle from the map, the system must redirect it to the application of the correspondent service (or open the link in the app store if the app if not installed in the phone).

G8. Guide the user through all the appointments’ locations as a GPS navigator.

R.0. The user must be registered and logged in.

R.35. When the user starts its travel, a GPS guide starts to run.

R.36. If a weather change is detected and it affects the track conditions, the system will propose a new itinerary.

R.37. If traffic is detected and it affects the track conditions, the system will propose a new itinerary.

R.38. If a delay of the selected ride is detected and it affects the track conditions, the system will propose a new itinerary.

R.39. If a delay by the user is detected and it affects the track conditions, the system will propose a new itinerary.

R.40. If a public transportation strike is detected and it affects the track conditions, the system will propose a new itinerary.

R.41. If the current position of the user is different from the one foreseen, the system will propose a new itinerary.

R.19. At the starting time of a trip, the system must verify if the previously computed track is still walkable to get to the appointment in time, checking:

* traffic conditions
* weather changes
* public transportation availability

R.20. At the starting time of a trip, the system must look for shared-vehicle and, in case, verify if they are a suitable alternative to the previous track and show them on the map.

c. User characteristics

* Not Registered user: a person using the application without being registered. The only available function is the possibility to proceed with the registration or to login in the system with a previously created account.
* Registered User: a person passed through a successful registration/login process. This type of user has all the functions available so it can create and fill its own calendar, personalize its account and schedule appointments.

d. Assumptions, dependencies and constraints

D.1 We assume that payment information correctness are verified using the PayPal service.

D.2 We assume that every username is unique.

D.3 We assume that the GPS retrieves the accurate position.

D.4 We assume that the registration’s email arrives to the destination without problems.

D.5 We assume that the weather forecast are always right.

D.6 We assume that, when an available shared-vehicle is selected, it remains available until the user completed the prenotation process.

D.7 We assume that the application always has an available internet connection.

D.8 We assume that all the web-services that the application uses to retrieve all the necessary information are always available.

**3. SPECIFIC REQUIREMENTS**

**A. External Interface Requirements**

**A.1 User interfaces**

//qui ci vanno i mockups

**A.3 Software interfaces**

This application is supposed to accomplish its task through the support of external services that collects data useful to efficiently arrange the trip of the users.

**Google Maps**

The system will connect to Google Maps, exploiting the relative APIs.

This is necessary to guide the user as a GPS navigator.

Furthermore, the Google service will provide all the data about the public transportation, the weather changes, the traffic condition and possible car accidents.

**OpenWeatherMap**

This service’APIs will provide all the weather forecast to optimize the travel planning along the day.

It will also be consulted every 5 minutes in real-time session to decide whether reschedule the travel or not.

**PayPal**

The PayPal API’s will be exploited to start a transaction towards the right public transportation service to let the user buy a one-ride ticket.

**Enjoy, Car2Go, Share’n’go, Ofo, Mobike, BikeMi**

The correspondent APIs of these sharing-services will be exploited to locate all the reservable vehicles.

If the user would like to reserve one of them, the system will redirect the user to the correspondent app or to the Play Store/Apple Store.

DBMS

The system is supposed to handle its databases through MySQL.

**D. Design Constraints**

**D.1. Standard compliance**

The system must ask the user to retrieve its GPS position.

The system will only use user’s email to validate its registration, therefore no SPAM email will be sent.

The system will not collect any credit card number, since the transaction to buy public transportation tickets is entirely redirected to PayPal.

**D.2. Hardware limitations**

Mobile App

- iOS or Android smartphone

- 2G/3G/4G connection

- GPS

A minimum 700MB of free space in the mobile storage is required.

**E. Software System Attributes**

E.1 Reliability

The system must do the scheduling and the other computing operations producing the correct outputs within 30 seconds.

E.2 Availability

The application and the relatives online databases must be available 24/7, with only few monthly hours of downtime permitted in case of updates or other maintenance operations.

E.3 Security

Users credentials and preferences will be stored using cryptography. The security and privacy of the communications between the application and the external servers are a primary concern: all the communications must use cryptography and the request of information to external services must be anonymous in order to protect the privacy of the user.

The user has to choose a secure password with a minimum length of 8 characters and it must be composed of numbers, symbols and mixed-case letters.

E.4 Maintainability

The system must be developed in order to make it easy to add new functions.

E.5 Portability

The application must support a wide range of mobile operative systems (at least Android and IOS).

**SCENARIOS**

SCENARIO 1: Registration and usage

Alessandro is a very busy business man and wants to organize his several appointments in order to know how and when reach them. His boss suggests him to download Travlendar+, so he signs up the app giving his username, password and email. Then he inserts in the calendar application all his weekly appointments and the system proposes an optimal solution to reach them in the minimum time last. He confirms all the application choices and becomes a very happy boy.

SCENARIO 2: Overlapping in two scheduling appointment

Andrea is a design student of Politecnico and has downloaded Travlendar+ scheduling his university calendar because he has to travel a lot to reach his classes. However her girlfriends wants to have a date on monday afternoon and Andrea add it in the app system. Unfortunately, on Monday he has a lecture at 12 o’clock near Bovisa, while his partner will wait for him in Famagosta at 14 p.m. So the system highlights the impossibility to have both the appointments. For this reason Andrea decides not to to university, by deleting the lecture from its calendar.

SCENARIO 3: Choosing preferences

Mario is a very green person, who prefers avoiding the usage of polluting transportation. After he registers for the planning application Travlendar+, the system asks him to insert some information in order to characterise his app account. For this reason he selects the ecologist preference, in particular he chooses bike as the favourite vehicle adding his personal bike in the correspondent section and, in the maximum foot distance option, he puts 1 km. Actually few days later he has to go to visit an Art museum situated in the other side of the city, and when he enters it as an appointment in the application, this one proposes him as the best alternative a bike track with his personal vehicle that lasts 2 hours and 17 minutes. Then, as second chance, the system suggests an itinerary with public transports of the duration of 1 hour and 36 minutes, which is shorter than the first one, but considering Mario’s preferences and the cost the application computed as not the best one. In fact Mario confirms the bike routes proposed and starts it.

SCENARIO 4: Considering weather conditions

John has scheduled his several appointments in the new Travlender+ app, selecting, inter alia, his preference in travelling by foot. However, for his business appointment on Monday there is a problem: rain is expected and the app is informed thanks to the support of OpenWeatherMap. Therefore, the system suggests him to use his own car, considering the type and the importance of the appointment too. John accepts the smart advice and confirms it.

SCENARIO 5: Considering appointments type and the priority

Steve is an important married businessman. Using his new Travlendar+ application he organised all his busy week with different type of appointment, such as a job meeting on Monday at 11 a.m., his yoga lesson on Wednesday at 7 p.m., his daughter’s concert on Friday (9:45 p.m.) and a date with his wife on Saturday (8 p.m). in a restaurant). He also adds the priority of all these events giving a 5 five stars evaluation to the meeting and to his romantic evening, 4 stars to the concert and only 2 for yoga. The system considers all these details and proposes him to use the personal motorbike on Monday because of its celerity in order to arrive at 9 o’clock, while for Saturday advises the usage of a car taking in consideration in this case the comfortability. On the other hand, the app suggests a walk to reach the concert 30 minutes earlier, actually because it so so distant from Steve’s home and finally proposes a bike track for Wednesday considering the fact that it is not necessary to be so tidy for this appointment.

SCENARIO 6: Real time updating

Sarah, who lives in Milan, wants to reach a start up convention in the afternoon and signed it in Travlendar+ app few days before, choosing a public transportation trip, suggested by the system in consideration of the fact that she has an ATM pass. However, an unexpected strike occurs and the application notifies the matter to Sarah. Sarah accepts the car alternative track and she begins the trip. Unfortunately, on the street he is going through, an accident happens, blocking the traffic and the system updates another time the itinerary. The system finds a bike sharing nearby and, considering plausible an arrival in time, suggests it to the user. She takes the bike and arrives right two minutes before the beginning of the convention.

EFFORT SPENT

Alessandro Saverio Paticchio: 4 hours

Andrea Tricarico: 4 hours

Davide Santambrogio: 4 hours

4 hours: 4 hours

CANCELLAREEEEEEE:

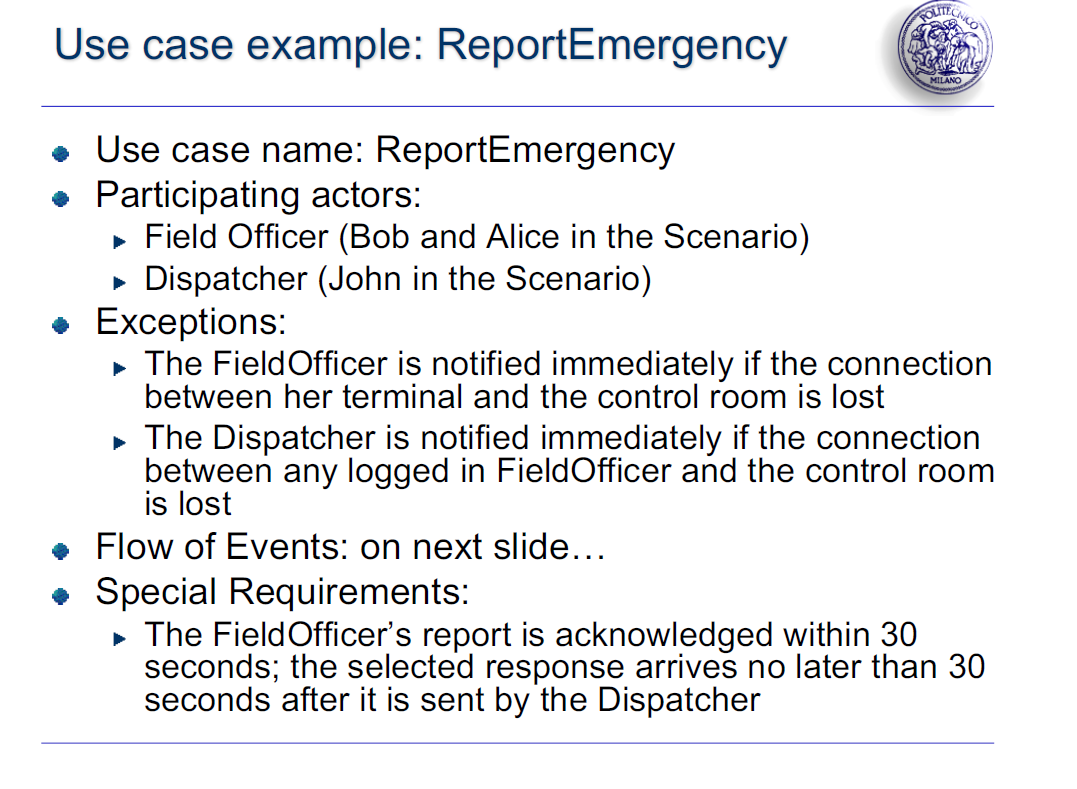
NOTA BENE:

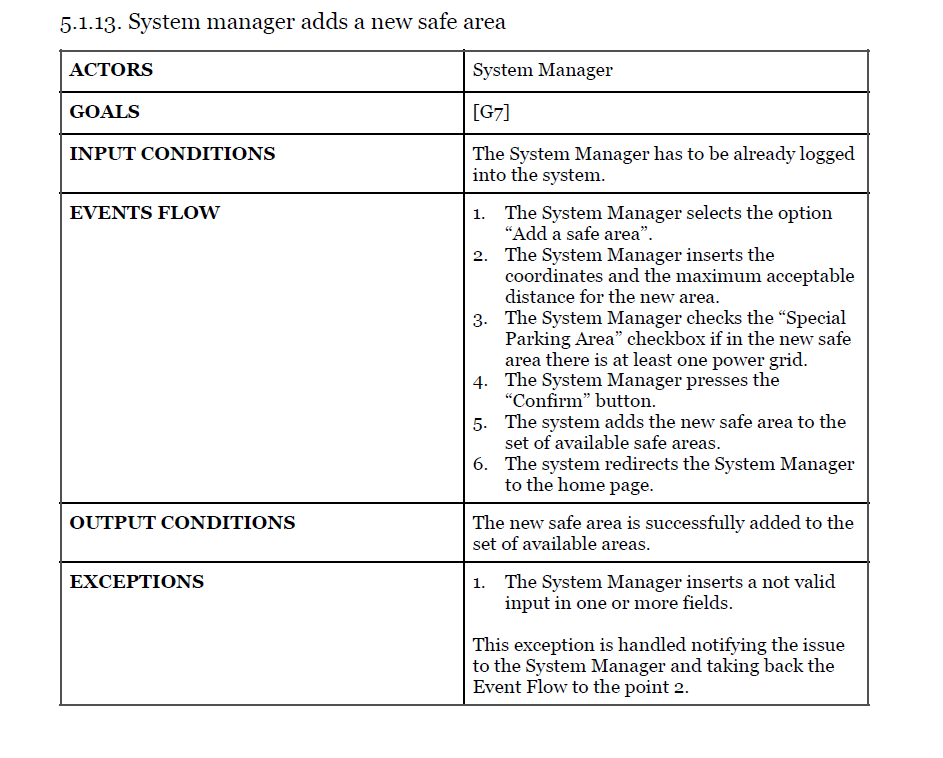
* controlla fase di pagamento, se non va a buon fine?
* cosa succede se non c’è connessione internet ? o gps? -> per ora nei domain assumption

AGGIUNGERE E MODIFICARE CHE TI FA PURE COMPRARE IL BIGLIETTO TRAMITE PAYPAL NELLE VARIE SEZIONI -> PER ORA STA SOLO NELLE DOMAIN ASSUMPTION

COSA è IL DOMAIN MODEL?

CONTROLLARE CONSISTENZA TRA TABELLA CON MINUTI DI ANTICIPO E SCENARI





|  |  |
| --- | --- |
| ACTORS | User |
| GOALS | [G3] |
| INPUT CONDITIONS | The user must be logged. |
| EVENT FLOW | 1. The user selects the option “Add appointment”; 2. The user completes the appointment’s description filling the following sections: Title, type, |
| OUTPUT CONDITIONS |  |
| EXCEPTIONS |  |