PROJECT PROP

SETUP DOCUMENT

Date: 27 September 2015

Group : E

Students: Ngoc Kieu Thanh Huynh – 2688093

Bilal Butt – 2688700

Michel Clerger – 2694646

Roman Siabro – 2224489

TABLE OF CONTENTS

PREFACE

This document contain information about the processes, requirements, the design of the website and application

Chapter 1: Processes for system events

Chapter 2: Processes, functional requirements and user interface for the website.

Chapter 3: Processes, functional requirements and user interface for the window application.

Chapter 4: Design for database

CHAPTER 1:
PROCESSES

The processes define all the major steps and functions involved in the organization of the event. Below we describe the processes from a sequential point of view. The processes outlined here define all the services and functions needed to make the festival event possible for the participants. The process are defined in the rough time sequence that they would occur in:

1. Registration:

Registration is the process whereby a potential participant in the event visits the website for the festival and makes the decision if he would like to register his/her intention to participate in the event. The website form would allow the visitor to register for the event and in return generate him/her a unique participant id which would also be emailed to them. Once the user has registered he would be prompted by the website to make payment then or at a later stage at the festival itself.

2. Payment:

As discussed before the user once having registered and making his intent to join the festival will be prompted to pay now by the website or he/she has the option to pay later at the festival event. If the user chooses to pay there and then at the website he/she will make the payment and an instance of his/her even account id would be created in the database, which is an entity in the database that holds the financial record of all the participants. One reason for having a separate participant id and a separate event account id for when the user pays his ticket fee is that participants who register on the website may not later want to come to the festival for whatever reason. Therefore the event account instance for the participant is only created once they make the payment for the entrance fee, which can be through the website or at the festival itself.

3. Camp Reservation:

Camp reservation according to the case can only be made in advance through the website. Therefore we made the camp reservation an option that the user can select only if he makes payment for his entrance fee on the website. Thus users wanting to pay the entrance fee later at the festival will not be allowed to make camping reservations. The reason behind this constraint is that users who have made the entrance fee payment in advance through the website are the ones who are more certain to actually participate in the festival and should be the one's allowed to make camp reservations. Users who have not paid their entrance fee in advance and would wait to pay it at the day of the festival should not be allowed to make camp reservations in advance because it is not entirely certain if they would even attend. The camping reservation would be made through the website once the user has made his/her payment for entrance fee from the website. In the camp registration form the user must enter the unique participant id's of his/her friends that wish to join him. The user will only be allowed to give the participant id of 5 other users who have already registered (notice: doesn't matter if those friends have not paid yet) as his/her co-campers. The user will then be returned an invoice for the camping reservation that he must pay at that moment. Payment for camping reservation cannot be paid later but must be paid there and then on the website.

4. Entrance to Event:

The entrance to the event will be based on the email that was sent to the users upon their registration. The user will present a hard or digital copy of the email with the unique participant id at the entrance desk and if the participant id exists in the database along with an event account id, which would mean payment has already been made, the user will be issued a unique passive RFID band corresponding to his/event account id. If the user has not made the payment yet and does not have an event account id but only a participant id he/she will be asked to make the payment at the entrance desk and once he/she has made the payment he/she too will be issued a unique RFID band and allowed to enter the festival premise. Each RFID will correspond to a unique event account id that contains information about the balance that the user has on his event account. He/she can use the RFID band to make purchases of food/drinks, borrow materials among other things with the RFID band. Also once the RFID has been issued to the user it will be scanned at the desk to indicate the users' entry to the festival.

5. Camp Entrance:

Camp entrance desk would be setup within the festival grounds and user can go to their camps once they are inside the festival grounds. There will be a similar entrance desk with an RFID scanner set up at the camping ground enclosure inside the festival ground. The database design for the camping is set-up in such a way that users who are co-campers of the person who made the reservation can enter their camp without the presence of the user who made the reservation. This possible because the database has an entity/table that assigns a group id to participants that are part of a group that has a camping reservation. Hence we can know which participants belong to a camping reservation through the unique group id that is a foreign key in the camping reservation entity as well.

6. Purchasing Drinks/Food:

Purchasing food and drinks would be made possible through the RFID bands that each user has been issued upon entry to the festival. The food and drink kiosks will have RFID scanners. The C# applications will be used at the food and drink kiosks where the vendor at the kiosk will generate an invoice through the C# applications which would indicate all the food items with their unique food id and quantity of each item purchased. Once the invoice with the sale amount has been generated the vendor will scan the users RFID to deduct from his/her balance the amount of the purchase.

7. Loan Material:

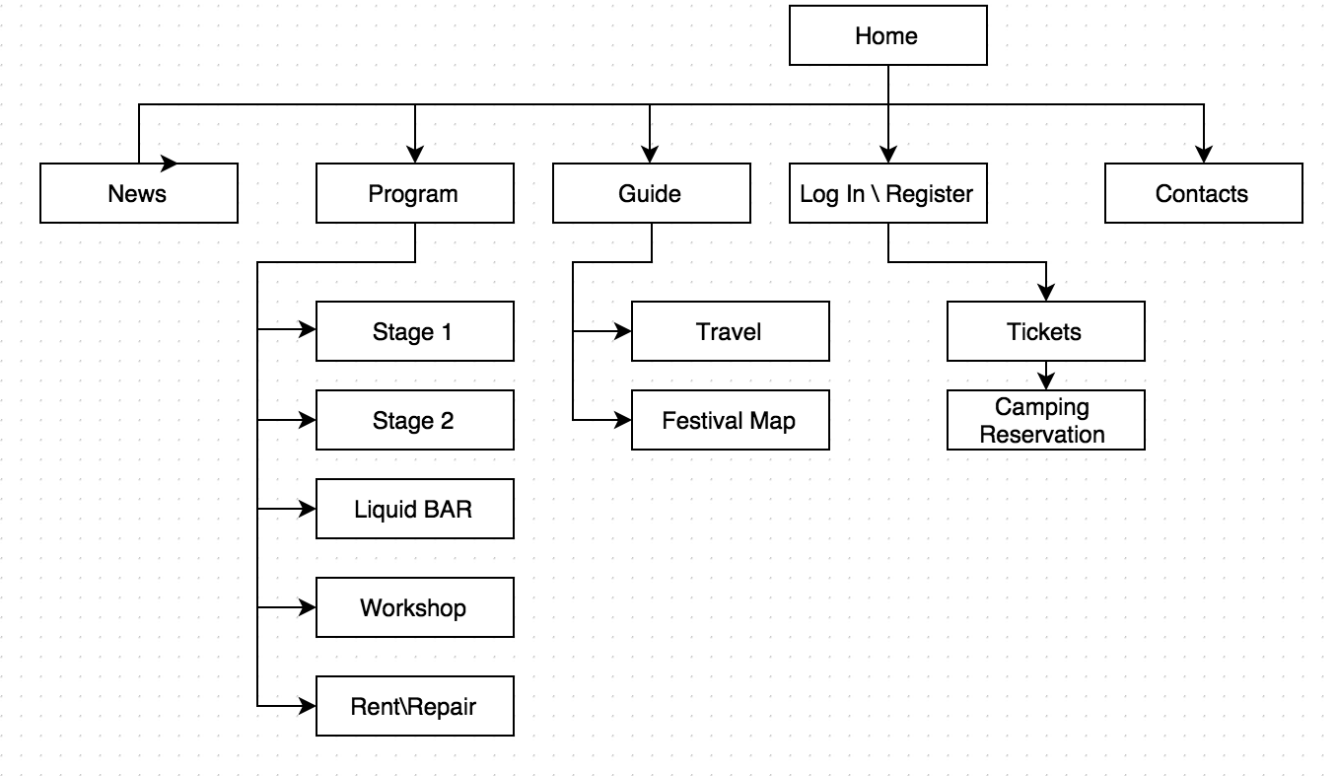
At the event, there is a stand that visitors can loan some materials like (laptops charger for Laptop, USB-cable, and camera). Visitors use the event-account to pay for it. They can loan more than one materials at one time and have to pay the deposit money for the materials that they would like to rent. They can rent for one or several days, and the loaning price is counted for the unit of one day.

CHAPTER 2:
WEBSITE DESIGN

Requirements and noticeable points for the website:

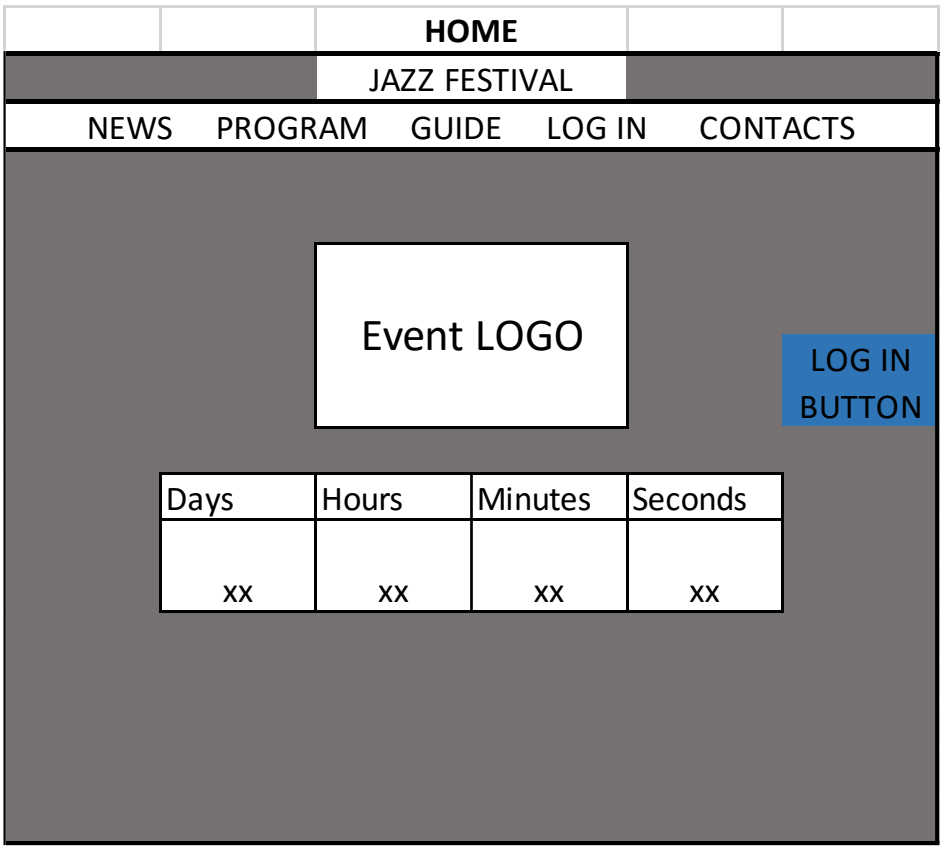
1. Who is the target audience of the website: the target audience is individuals interested in Jazz music, therefore the target demographic is quite large because people of various age groups would be interested in Jazz music
2. Goals of the website:
   - to advertise the event among the audience
   - provide information regarding the specific events and musicians playing at the festival
   - to allow participants to register and pay for the entrance of the event
   - to allow participants, if they wish to rent a camping ground, to sign up and pay for a designated camping area on the festival grounds
   - provide contact information for users of website if they need more information
3. Key information for visitors:
   - time, location and performances at the festival
   - how to register by website for the event and make payment, website must clarify that participants MUST register before the start of the event but can pay at the event at an extra costs of 10 euros if they register at the event

Site map

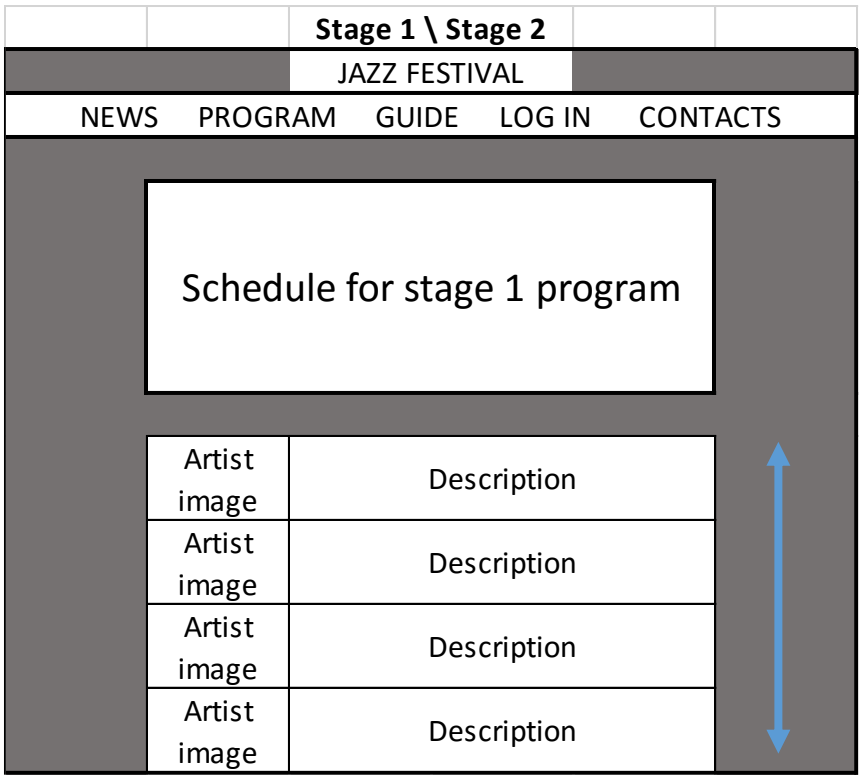


Wireframe and description for website:

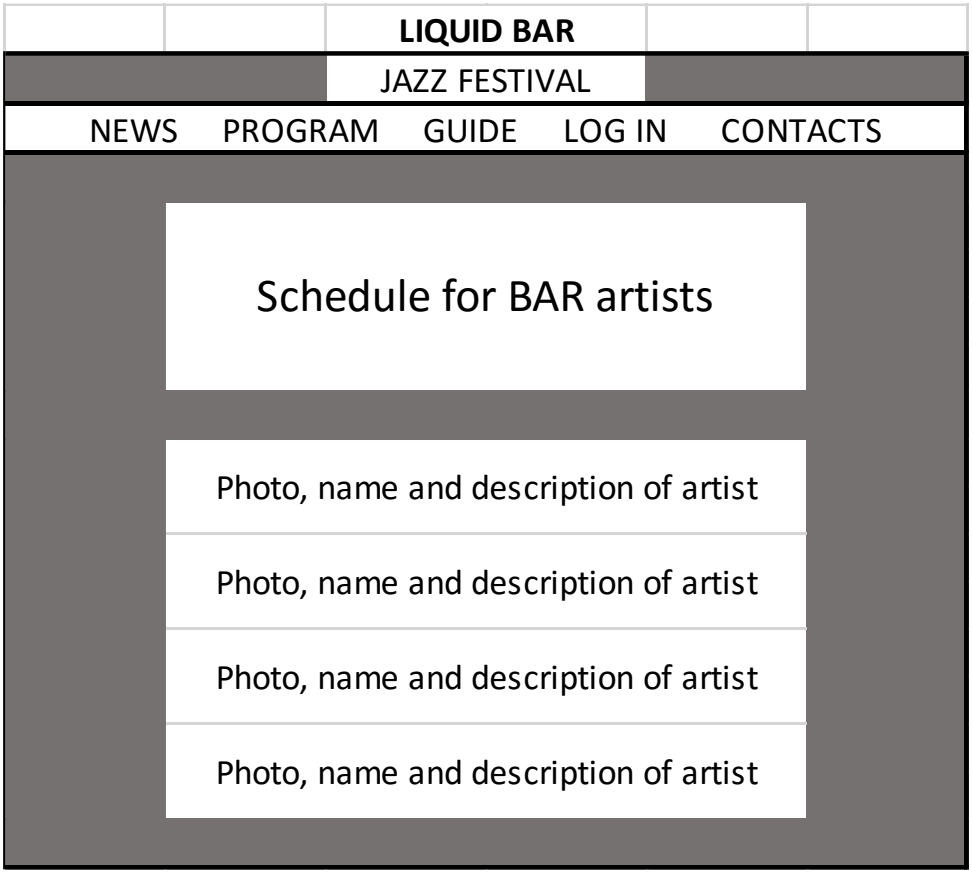
- Home:
  - On the homepage the logo of the event will be displayed also as the exact time (days/ hours/ minutes/ seconds) until event takes place.
  - There will be also a Log in button in order visitors can immediately proceed to buying a ticket or renting a camping place.



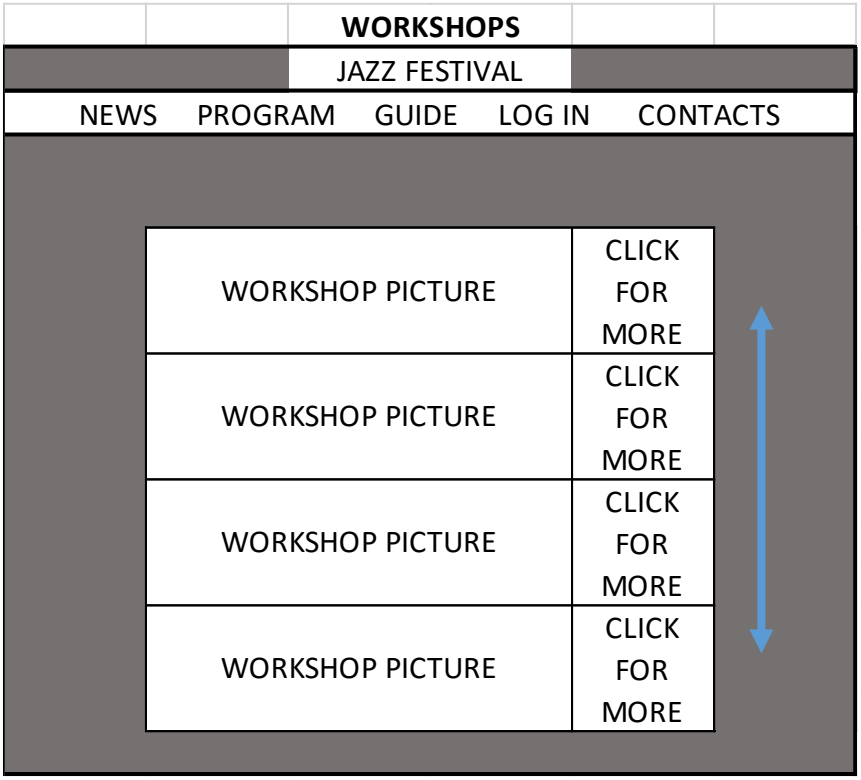
- Stage 1 \ Stage 2
  - On this page visitor can see the timetable where he/she can check who and when is going to perform. Below you can find a description for each artist that is going to perform on that stage.


Stage 1 \ Stage 2
JAZZ FESTIVAL
NEWS PROGRAM GUIDE LOG IN CONTACTS
Schedule for stage 1 program
Artist image
Description
Artist image
Description
Artist image
Description
Artist image
Description

- Liquid BAR

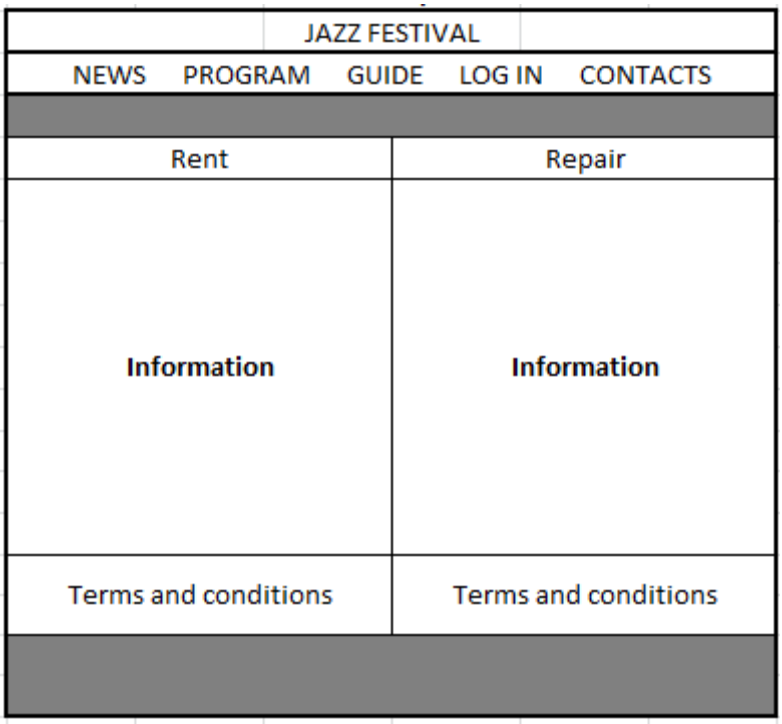

LIQUID BAR
JAZZ FESTIVAL
NEWS PROGRAM GUIDE LOG IN CONTACTS
Schedule for BAR artists
Photo, name and description of artist
Photo, name and description of artist
Photo, name and description of artist
Photo, name and description of artist

- Workshops
    - Here the pictures for different activities will be displayed. By clicking a button (More), visitor can find more information about exact workshop.

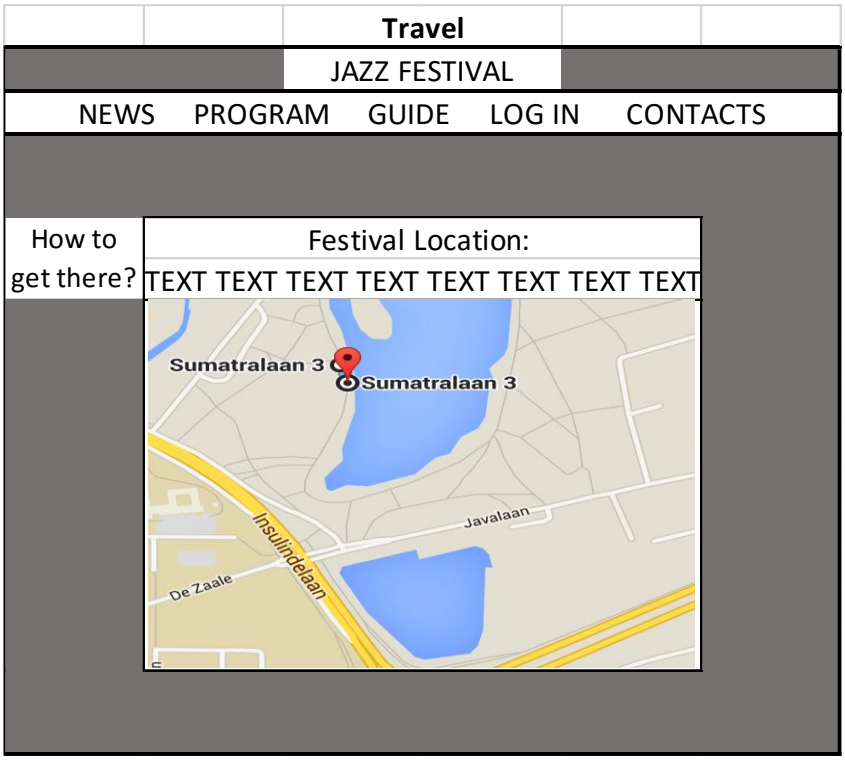


- Rent \ Repair

On this page we will provide information about rent and repair facilities during the event also as terms and conditions of using our service.

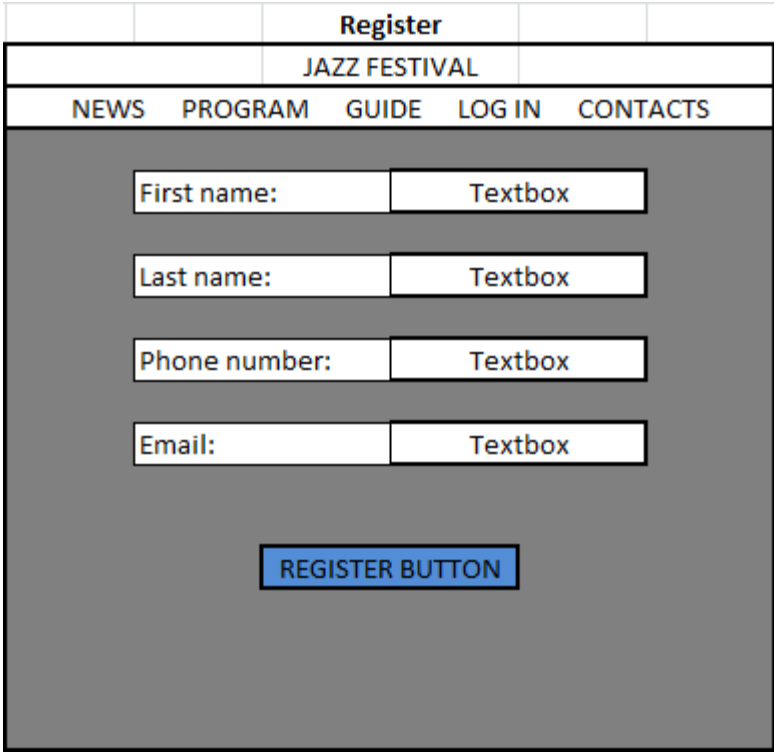


- Travel
  - On travel page visitor will find an exact address of the event. Here he will be able to plan a trip from any point by using the google maps.



- Festival Map
  - On this page visitor will find a map of the event with all the information regarding stages, camping places, snack bars, repair shop and etc.

- Log in \ Register



- Tickets

TICKETS

JAZZ FESTIVAL

NEWS PROGRAM GUIDE LOG IN CONTACTS

DAY TICKET - THURSDAY- €49.00	Buy button
DAY TICKET - FRIDAY- €49.00	Buy button
DAY TICKET - SATURDAY- €49.00	Buy button
2 DAY FESTIVAL PASS (THU - FRI)- €69.00	Buy button
2 DAY FESTIVAL PASS (FRI - SAT)- €69.00	Buy button
3 DAY FESTIVAL PASS (THU - SAT)- €99.00	Buy button

- Camping Reservation

CAMPING RESERVATION

JAZZ FESTIVAL

NEWS PROGRAM GUIDE LOG IN CONTACTS

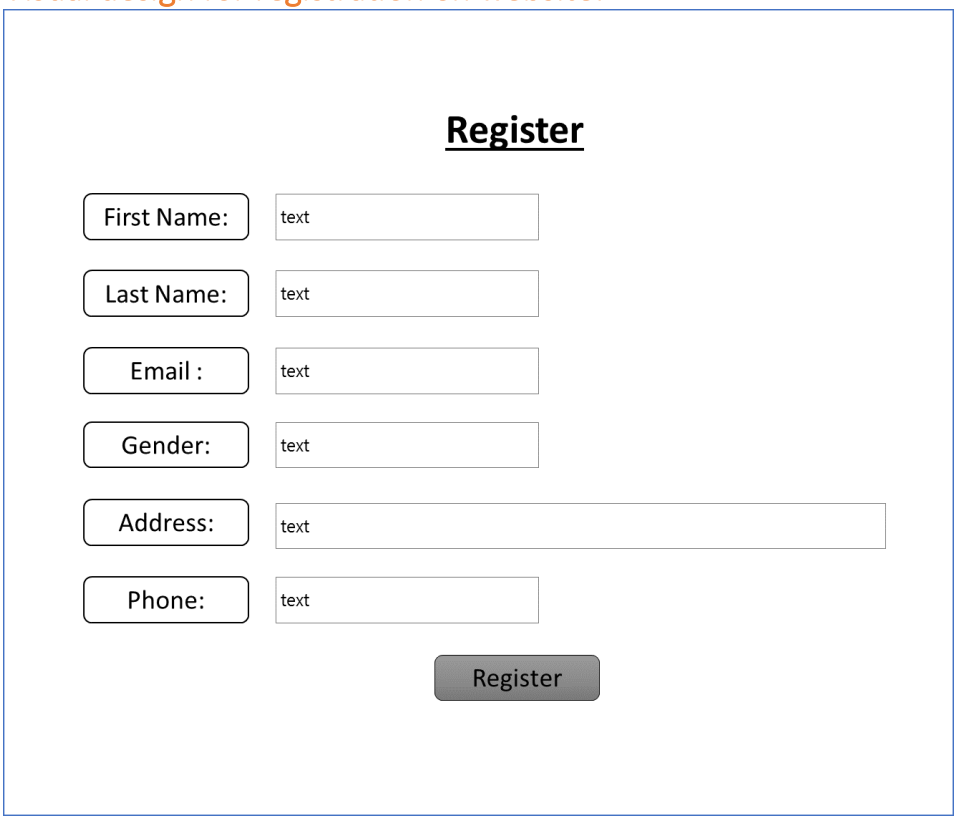
Camping place image	(VIP camping) Description	Buy button
Camping place image	(Regular camping) Description	Buy button
Camping place image	(Cottage camping) Description	Buy button
Camping place image	(Car camping) Description	Buy button

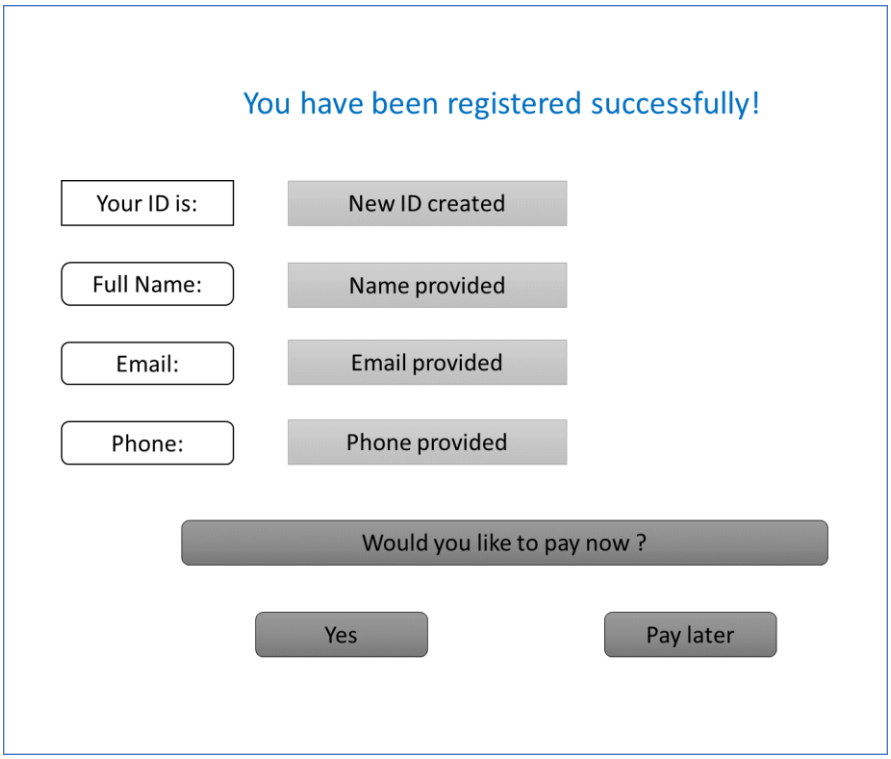
- Contacts
    - On Contact page visitor will have to fill in their personal details in order to contact event co-ordinators.

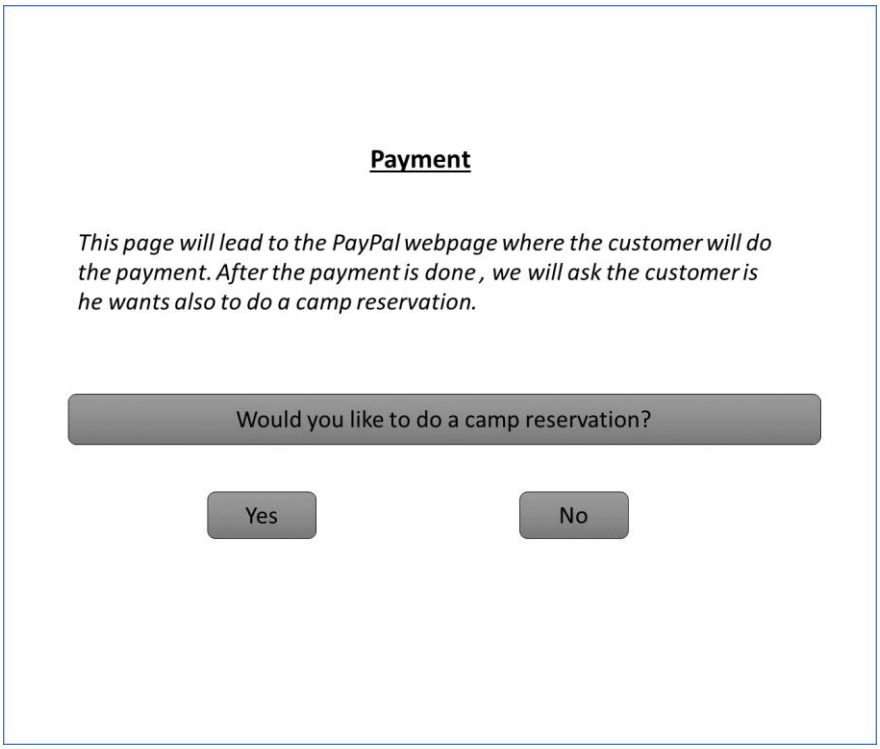
MoSCoW method for website's pages:

Functions/features	MUST	SHOULD	COULD	WILL NOT
Countdown clock until the event (Home)	X			
Schedules for stages/ bar			X	
Check artist information			X	
Workshops information			X	
Rent and Repair page (prices, terms and conditions)	X			
Check festival location/ route		X		
Festival map	X			
Account registration		X		
Log in / log out		X		
Book a ticket(s)	X			
Book camping spot	X			
Contact festival administration	X			
dynamic site: twitter (client and/or server side)			X	

Visual design for registration on website:




You have been registered successfully!
Your ID is:
New ID created
Full Name:
Name provided
Email:
Email provided
Phone:
Phone provided
Would you like to pay now ?
Yes
Pay later


Payment
This page will lead to the PayPal webpage where the customer will do the payment. After the payment is done , we will ask the customer is he wants also to do a camp reservation.
Would you like to do a camp reservation?
Yes
No

Camp registration

Member #1:	text
Member #2:	text
Member #3:	text
Member #4:	text
Member #5:	text
Days:	From : - To :

Register and pay

Payment

This page will lead to the PayPal webpage where the customer will do the payment. After the payment is done , we will send an email to the customer with a receipt for his payment and details about his registration!

CHAPTER 3:
WINDOW APPLICATION DESIGN

The application is used at the entrance of the events

1. THE REQUIREMENTS OF ENTRANCE APPLICATION

At the entrance of the events, the organizer have a demand to check if a visitor is allowed to enter or not base on their registration on the website. This one will be check by their identification via Bar-code or QR-Code or RFID chips.

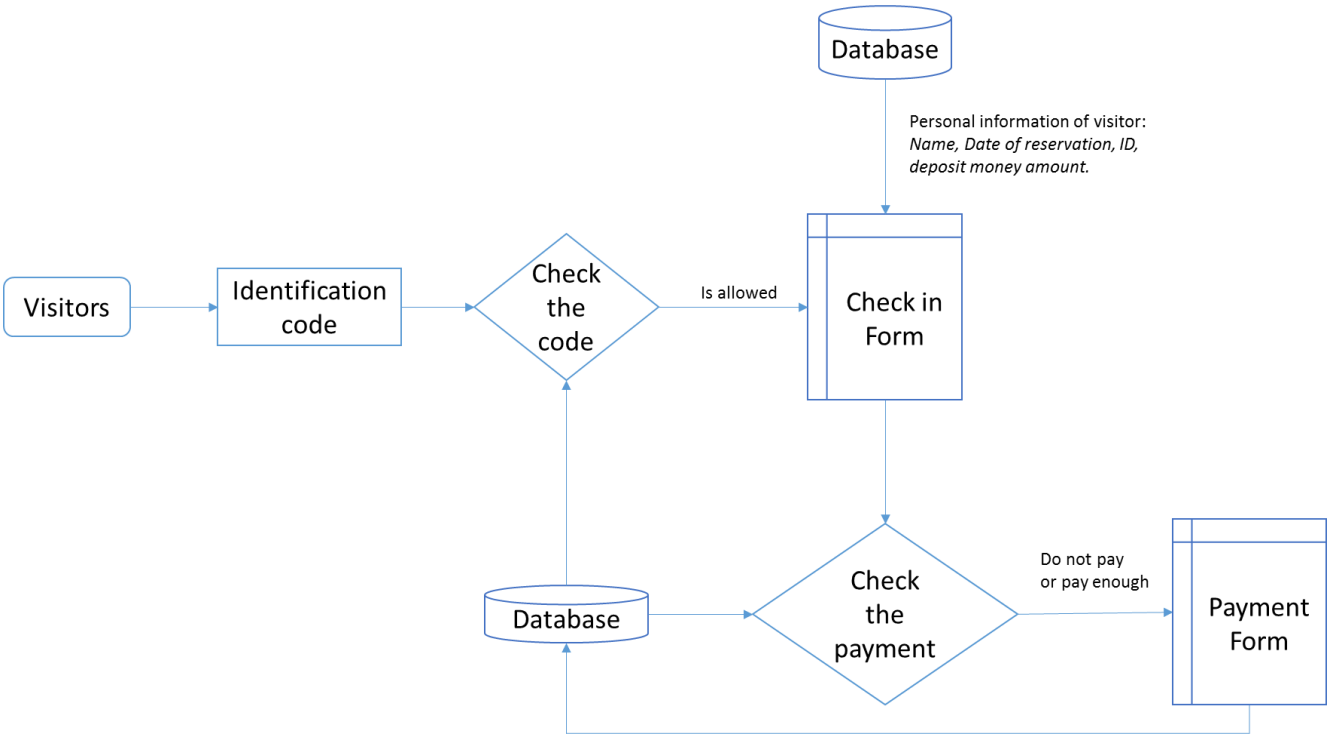
Besides, the organizer need to check the payment of visitors. If they did not pay or did not pay enough, they should get the possibility to pay.

2. FUNCTIONALITIES

Base on the requirements, the functionalities for this application be suggested in this table, following Mos-cow methods:

Functionalities	M	S	C	W
Check the allowance of the visitors using RFID	X			
Check the payment of visitors	X			
Allow visitors make the reservation at the entrance of the event		X		
Allow visitor paid more money by cash if they did not pay or did not pay enough			X	
Allow visitor paid online	X			

3. PROCESSES



Finger 2.1 Entrance process

4. ID SYSTEM SELECTION:

Participants in the festival would need to have an identification system in order to enter/re-enter the event premises, make payments for food or beverages and borrow materials as well as other such activities. We have proposed the use of either QR codes or RFID tags for such identification of the participants. We will discuss briefly the pros and cons for the two identification methods that we can use for our project: QR codes and RFID.

QR Code:

QR code is a 2-dimensional barcode, also called a matrix barcode, which can be read by an imaging device such as camera or a scanner. The code is made up of black modules placed on a square grid with a white background.

Pros:

- can be read by any imaging device, so easy setup
- can be read from phones or printed pages held by the festival participants, thus is easy, cheap and convenient for the users
- tend to be quite reliable

Cons:

- malicious QR codes can be used to harm the host system being used for scanning the code

RFID:

RFID (radio-frequency identification) uses electromagnetic fields to transfer data and is used for identifying and tracking tags attached to objects such as wristbands. The tag may be active or passive meaning in which case the active tag requires a small battery whereas the passive tag does not. RFID require a specific device called tag readers that can read data transmitted from the tags. In our case we are more likely to use passive tags with Active Reader Passive Tags called ARPT's.

Pros:

- are very reliable
- the setup and devices needed for this system are available at ISSD

Cons:

- would have to get the tags made separately for event participants which may be expensive
- would need to figure out how to deliver the tags to the event participants before the start of the event
- what actions would need to be taken if users lose their RFID tags

Camping Reservation Application

1. REQUIREMENTS OF CAMPING RESERVATION APPLICATION:

The process for checking in at the camping sites would be similar to as checking into a conventional hotel. To make this process possible we will be developing a camping reservation app that would help keep the organizers of the festival to keep track of visitors reservations for camp sites and if they have in fact checked into the camp sites.

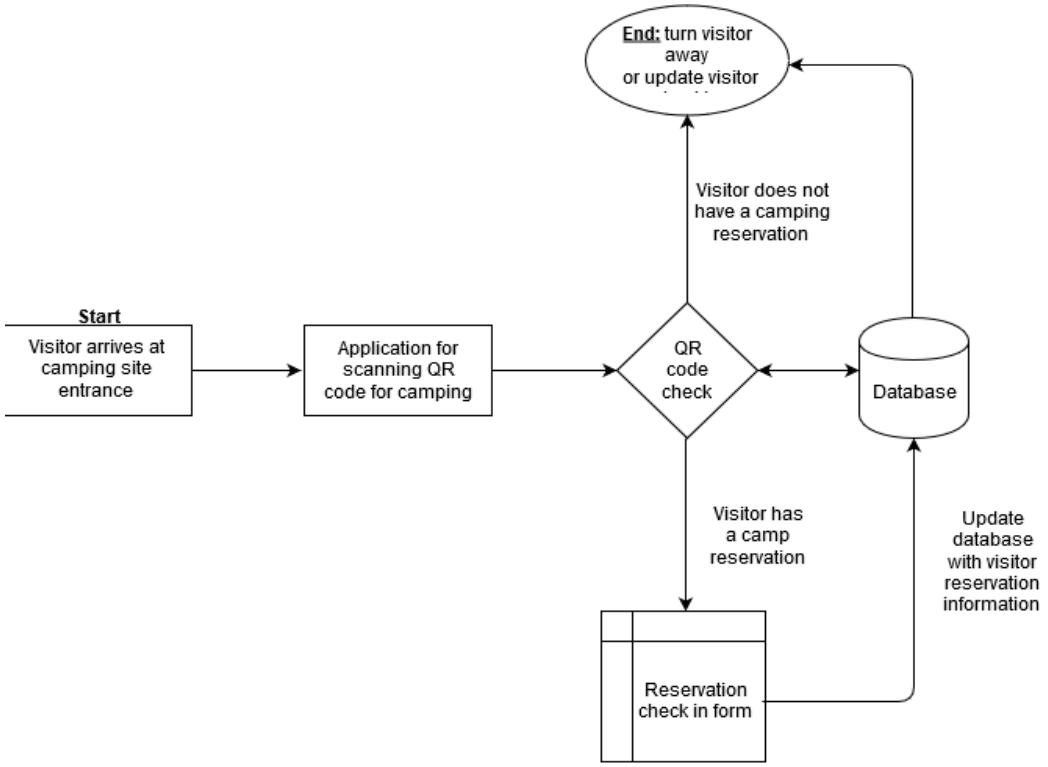
The camp reservation process as a whole, not just the computer application, would work on a similar basis as hotel check in work. A client or festival participant in this case would make a reservation in advance mainly through the website and his/her reservation would be on file in the database. Then when the client subsequently visits the festival he/she will check into the camp site. The camp site would have entrance desks set up where the visitor would be allowed to enter only if they had reservations and check in as guests based on those reservations.

2. FUNCTIONALITIES:

Base on the requirements, the functionalities for this application be suggested in this table, following Moscow methods:

Functionalities	M	S	C	W
Scan QR code of visitor to check for reservation	X			
Allow visitor to make camp reservation on spot			X	

3. PROCESSES



Finger 2.2 Camping reservation process

The application is used at the shops for food and drinks:

1. THE REQUIREMENTS OF SHOPS APPLICATION

At the event, the organizer have a demand to sell food and drinks for visitors at the stands on the event's area. For pay desk, they need an application being able to generate the sale receipt. The payment is paid by the event-account which is check by identify number.

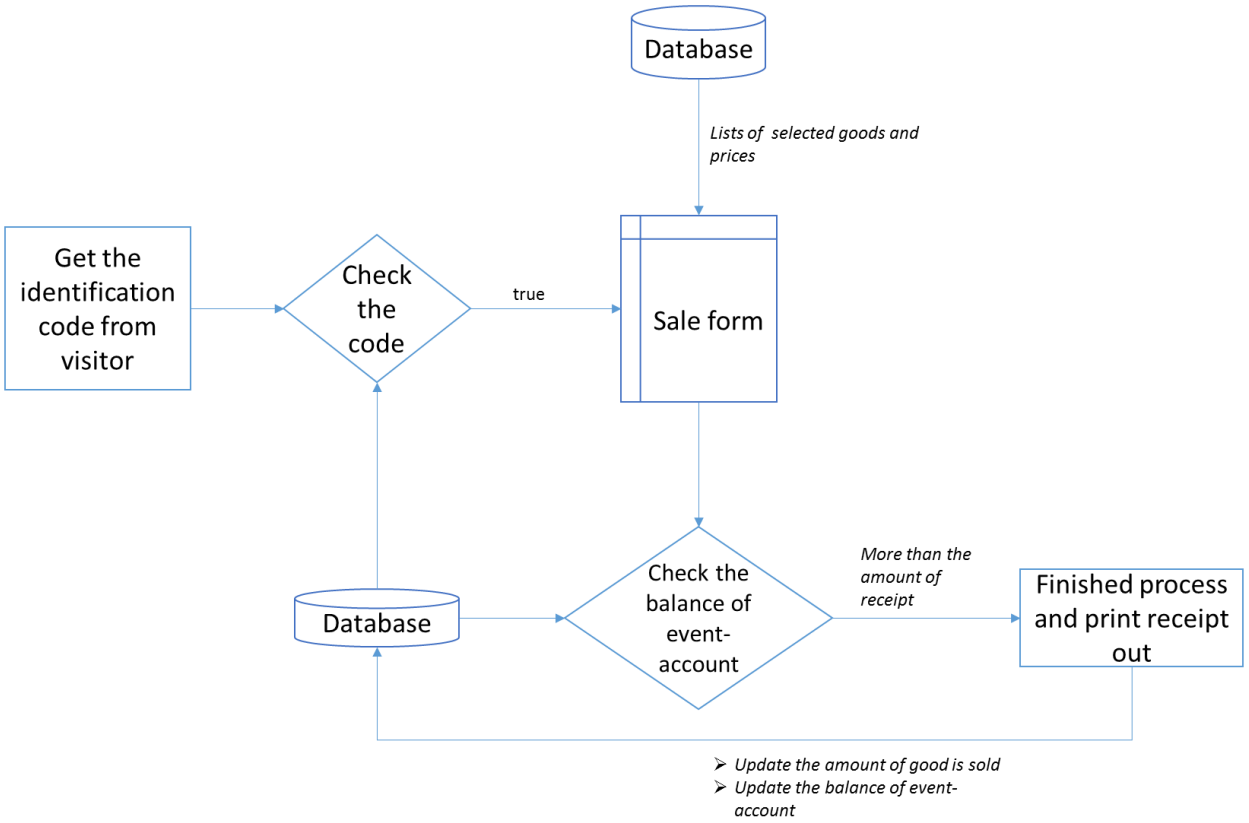
Besides, the organizer need to check amount of receipt have to less than the balance of the event-account.

2. FUNCTIONALITIES

Base on the requirements, the functionalities for this application be suggested in this table, following Moscow methods:

Functionalities	M	S	C	W
Check the event-account using identify number	X			
Generate the sale receipt.	X			
Check the balance of event-account (must be higher than the amount of sale receipt)	x			
Check the amount of available for each type of food and drink and allow update the once is out of stock.			X	

3. PROCESSES



Finger 2.3 Process for shop application

The application is used at the stand for loaning materials:

1. THE REQUIREMENTS OF LOANING MATERIALS

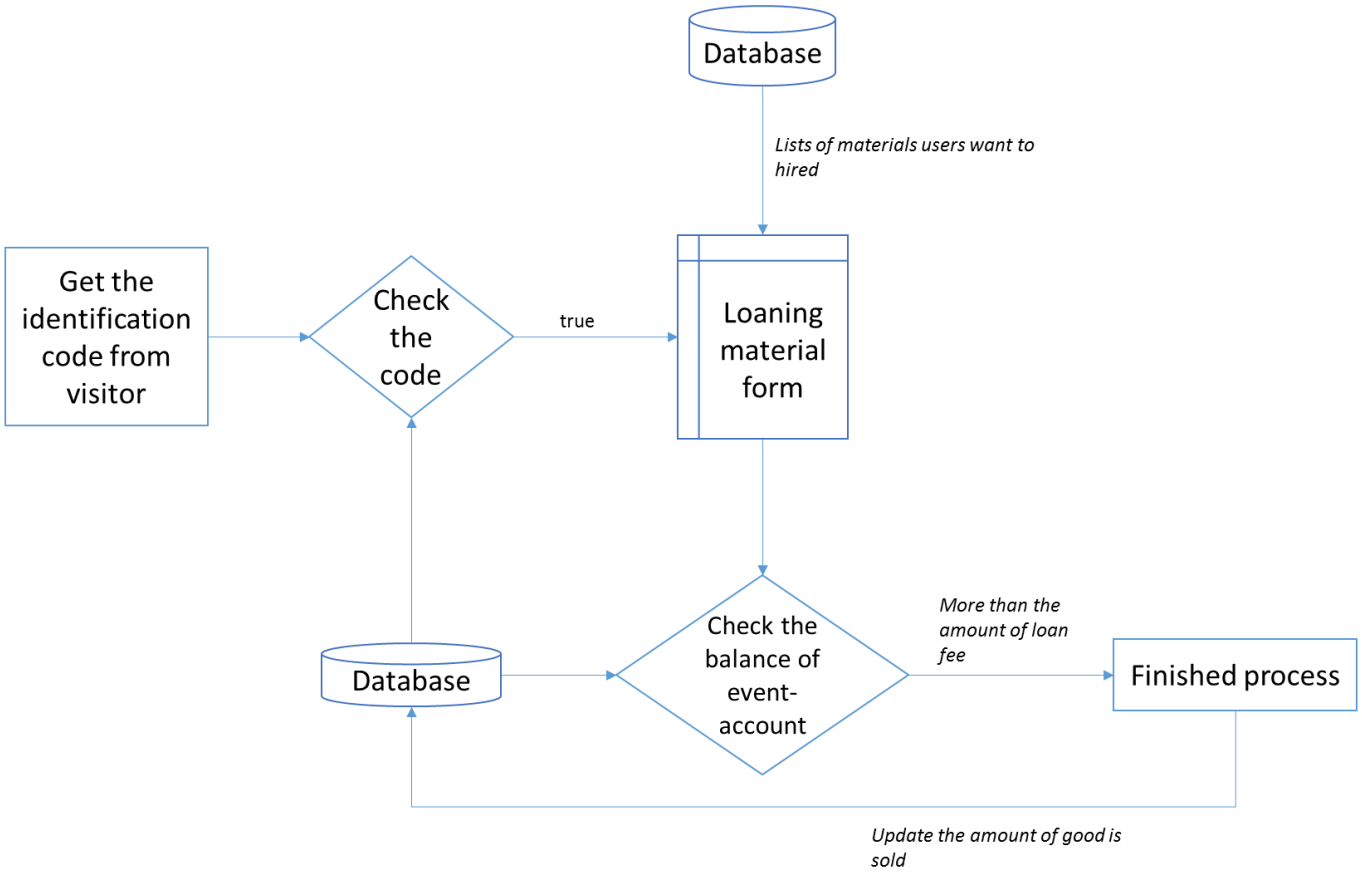
At the event, there is a stand that visitors can hire some materials like (laptops charger for Laptop, USB-cable, and camera). Visitors use the event-account to pay for it.

2. FUNCTIONALITIES

Base on the requirements, the functionalities for this application be suggested in this table, following Mosco methods:

Functionalities	M	S	C	W
Check the event-account using identify number	X			
Paid for loan fee using event-account	X			

3. PROCESSES



Finger 2.4 Process for shop application

C# Application for checking the status of the event

In order to design an application that would give feedback on the status of the event to the event organizers, we need to understand the kind of relevant information that the organizers of the event would be interested in obtaining.

We make several assumptions about the information that the organizers would want from such an application. Below we list a set of relevant information that the event organizers would be interested in knowing in order for the smooth running of the event.

EVENT INFORMATION REQUIRED

- *Capacity:* number of people currently on the event grounds, total number of people that have signed up for the event and purchased tickets, number of people checked in at camp sites
- *Inventory:* inventory of food and drinks at the event stalls
- *Financial:* financial information about ticket sales, food/drink sales, camping site revenue, and loan material revenue

PROCESS FUNCTIONAL REQUIREMENTS

We identify three possible reporting areas that the event organizers would be interested in having regular reports about. These areas concern the capacity, inventory and financial aspects of the event.

The application that we will design for this process will allow users to assess the following information from each of three reporting areas *capacity, inventory and financial*:

Capacity:

The report on capacity is intended for the organizers to manage any capacity related questions they might have about the event. It would be useful for instance for the organizers of the event to know the total number of people expected at the event which could be found out by the number of unique registrations that have been made from the website for the event. Also it may be important for the organizers to know how many people have currently joined the event in order to make decisions about managing the event better. Based on our assumptions of the relative importance of each information we have assigned Moscow rating for each field of information that the C# reporting application should implement.

Field in Report	**Description of field**	**Moscow**
Registrations	Is the number of registrations for the event through the website	M
Purchased Tickets	The number of tickets purchased for the event	M
People currently at event	Number of people currently checked into the event	M
Camping sites rented	Number of camping sites rented out	M
Campers checked in	Number of campers checked in	S
Total campers	Includes all campers those checked in and those still to come	S

Inventory:

The inventory report would be classified as a WOULD under the Moscow framework, because the C# application to be used at the drink/food shops would already have capabilities to report on the inventory levels. Therefore it would be redundant to include an Inventory sub-report in the reporting application for the event that same information could be found from the C# application designed for the food/drink shops.

Field in Report	Description of field	Moscow
Number of Items	The number of items different items that are kept in inventory	M
Inventory Item 1	Quantity of the item 1	M
…	…	
Inventory Item n	Quantity of item n	M

Financial:

All the fields in the financial sub-report would be essential for the event organizers to assess the revenue streams they have generated from each activity of the event. Therefore all the fields in the Financial sub-report are a must have according to the Moscow framework.

Field in Report	Description of field	Moscow
Ticket sales	Revenue from ticket sales	M
Food/Drink sales	Revenue from food and drink sales	M
Camp site revenue	Revenue from camp site rents	M
Loan material revenue	Revenue from stall for loaning materials	M

Pay-Pal Application

1. REQUIREMENTS OF PAY-PAL APPLICATION:

This applications purpose is to convert a PayPal-text-file that would update the database about the various transactions that would occur through PayPal. The application would allow the users, in this case the festival organizers, to upload a file to the application and translate that data to update the database in a relevant manner about the financial transactions that are recorded on the text file.

The text-files would be delivered from time to time by PayPal to the festival organizers and the purpose of this application would be to update the database regarding the transactions that would be recorded on the text file. The following is a Moscow matrix for the application that outlines the functional requirements of the application

2. FUNCTIONALITIES:

Based on the requirements the following are the functionalities of this application:

Functionalities	M	S	C	W
Convert pay-pal text-files to meaningful information regarding customer transactions(use Event-account id's for identification of customers in pay-pal transaction)	X			
Translate the information in the pay-pal text-files and display to rich text box or list box in the application		X		
Notify users by e-mail or other method of receipt of their payment		X		
Update database table of Event-account by updating the balances of the visitors(based on their event-account id) who have made payments according to the pay-pal text-file	X			

USER INTERFACE DESIGN

1. MAIN FORM OF THE APLICATION

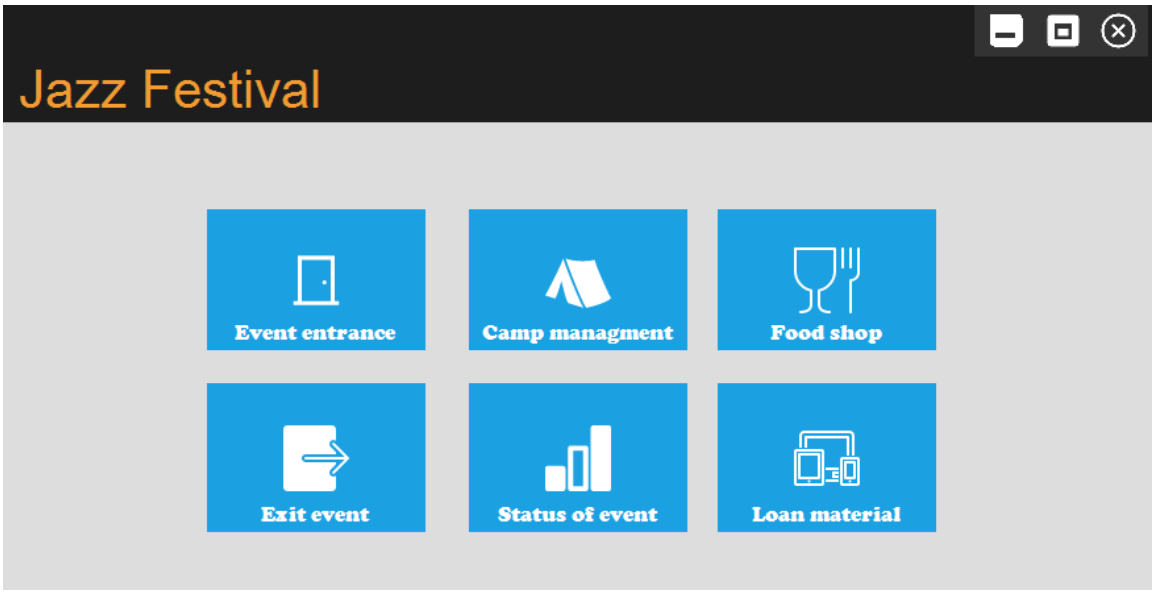
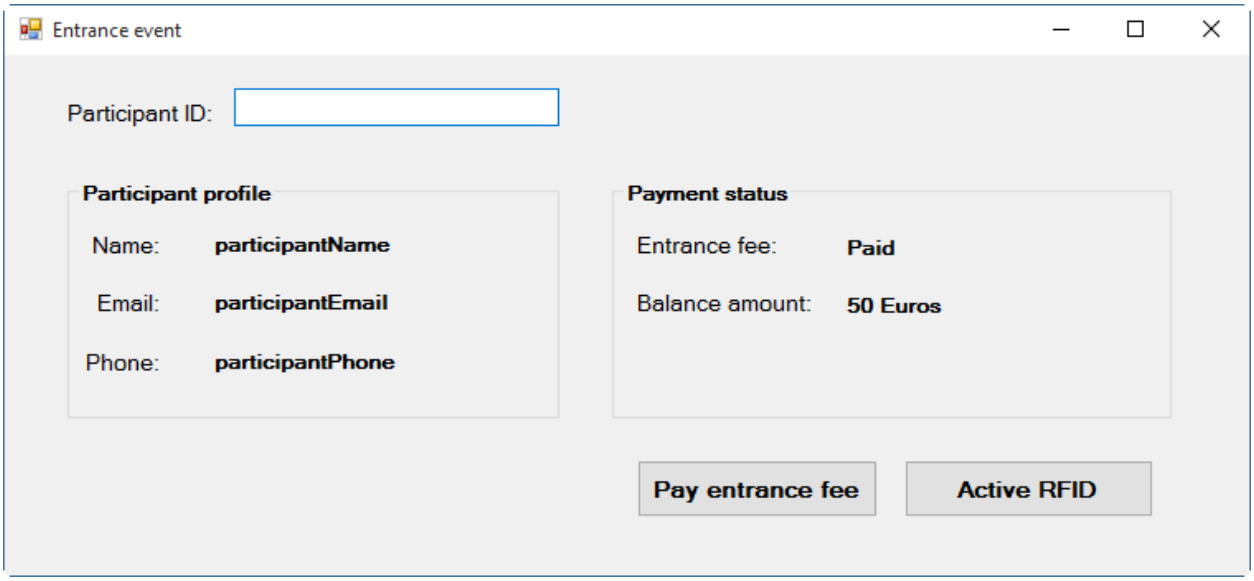


Figure 2.5 Main form UID

This form provides the functionalities to link to 6 main features of this application. The design style base on the Metro design principle.

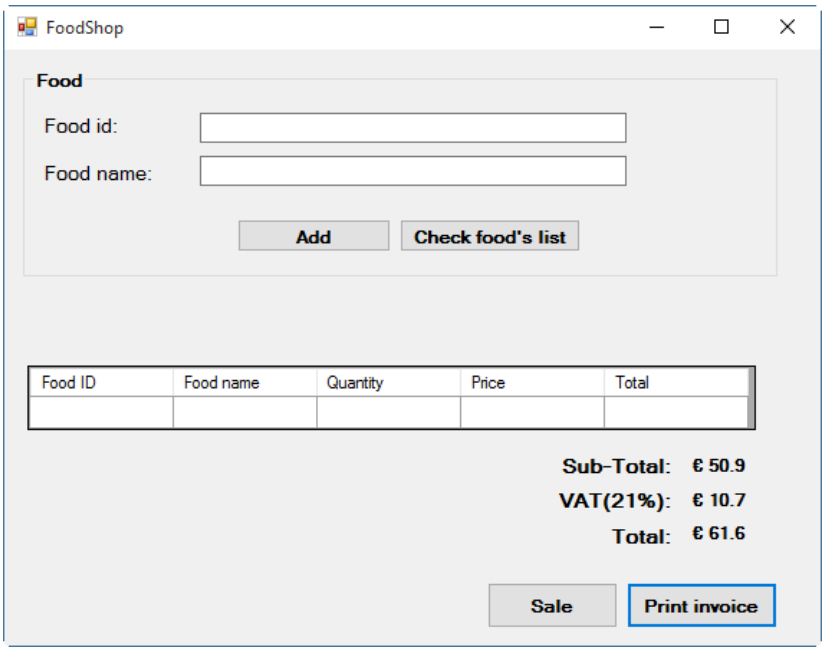
2. CHECK IN AT THE ENTRANCE OF EVENT FORM



When the joiners come to the event, the will give their participant ID, which was confirmed by email, the application will check their ID, load their personal information and payment status. In case of they did not pay the entrance fee, they have to pay there. The organizer allow them to pay by click the "pay entrance" fee button If their balance amount is zero and they have not pay yet, they have to use PayPal to transfer their money into the account of the event's organizer.

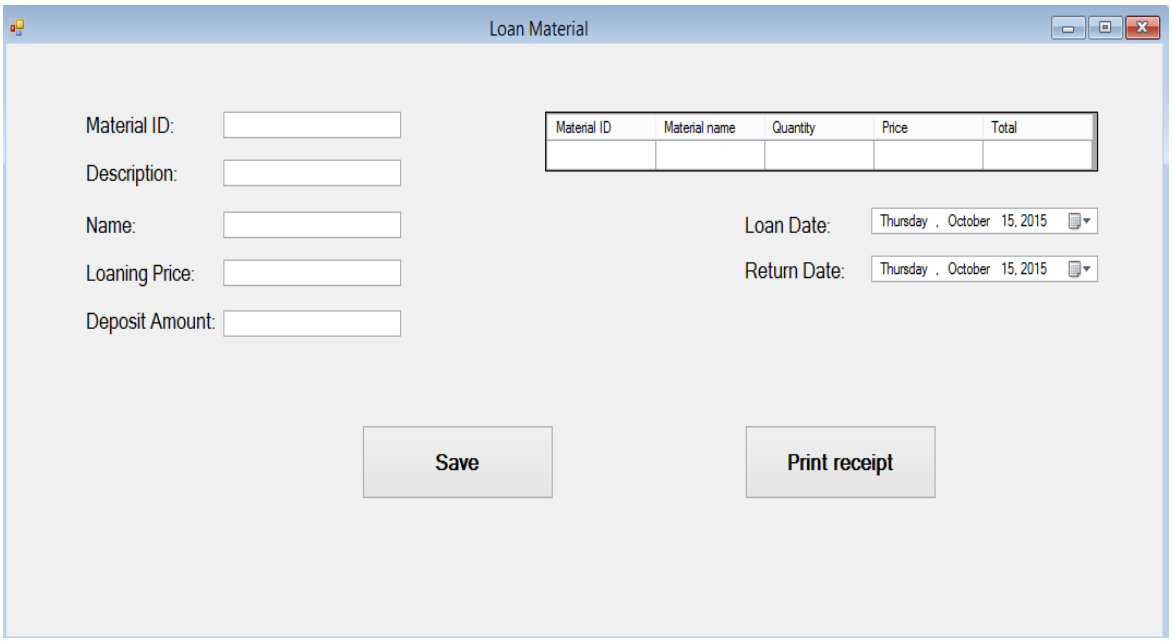
If all of payment is done, they will be provided an RFID bracelet to identify their event-account during the event.

3. FOOD SHOP APPLICATION



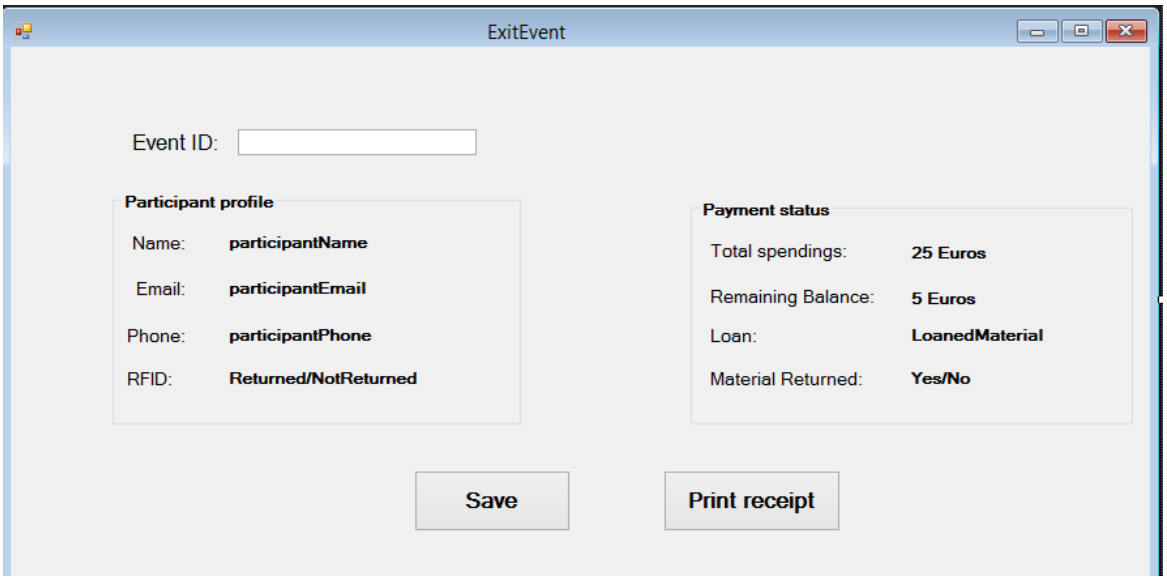
Purchasing food and drinks would be made possible through the RFID bands that each user has been issued upon entry to the festival. The food and drink kiosks will have RFID scanners which scan their RFID bracelets when they buy. First the sale clerk will type in the food id and the food name will be displayed. In case of they do not remember the food id, the can click on the "**check food's list**" button to see the list of all inventory. After doing that, they choose the quantity of the item. If they type the same food_id, the quantity will be increase by one. When they click sale, the participant will scan their code and the payment will be done. If the customers would like to receive the invoice. The "Print invoice" is used to do this task.

4. LOAN MATERIAL



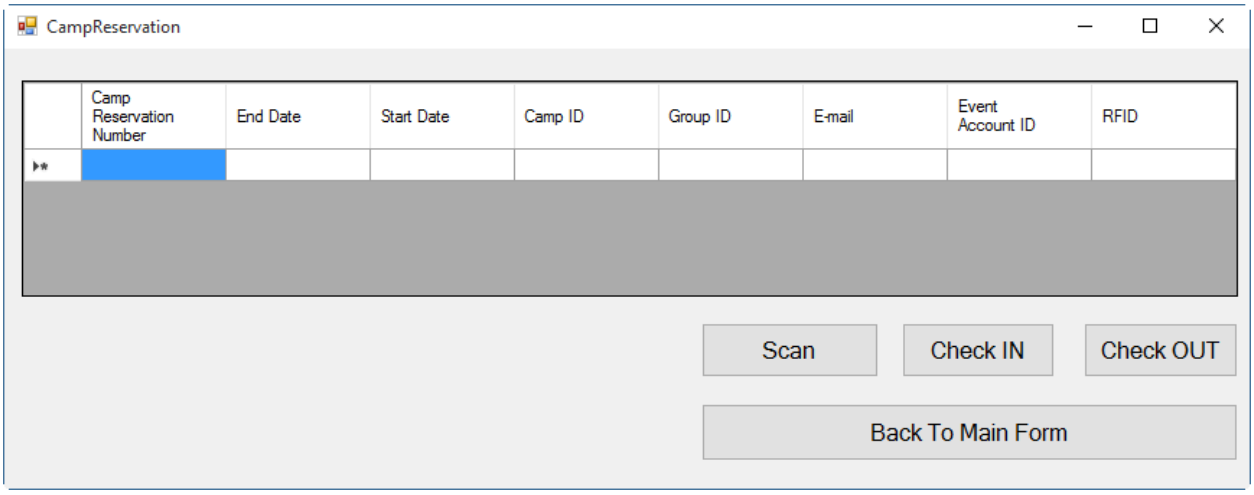
We will keep track of the loan materials with that application. We will check the ID of the material, we will also save the names, description and also the number of materials that the guest want to loan. And we will save the date of the loan. Once the products are returned we will then add the return date, otherwise it will mean that the materials have not been return. We can then print a receipt for when the item is loaned with every specification of the materials and also when it is returned as a proof that the guests returned their materials.

5. EXIT EVENT



At the end of the festival when the guests leave, it will be possible for them to check the amount of money they have spent and their remaining balance. They can also print it with their personal details in case they want to keep it as a proof or a reminder. This can be useful in case of claims or if a customer had any trouble with their spending during the festival.

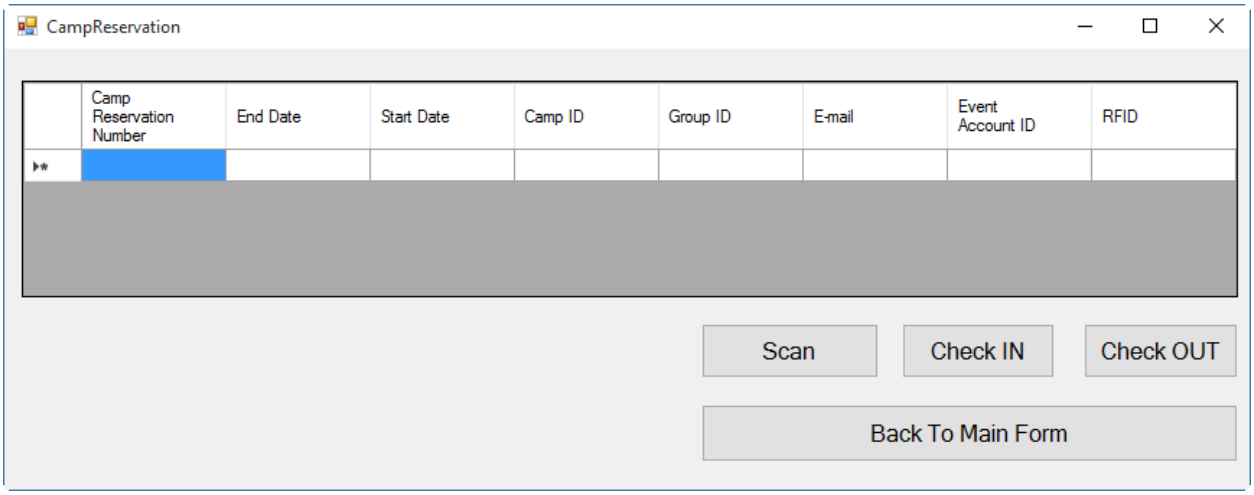
6. CAMP RESERVATION APP



The camp reservation app would work by the event staff scanning the RFID bracelet of the entrant. If the RFID is recognized by the database the app will show the complete camp reservation details of that person's group in a data grid view on the app. The row of information pertaining to the person whose RFID has been scanned will be highlighted in a different color (yellow) on the data grid view to indicate who that person.

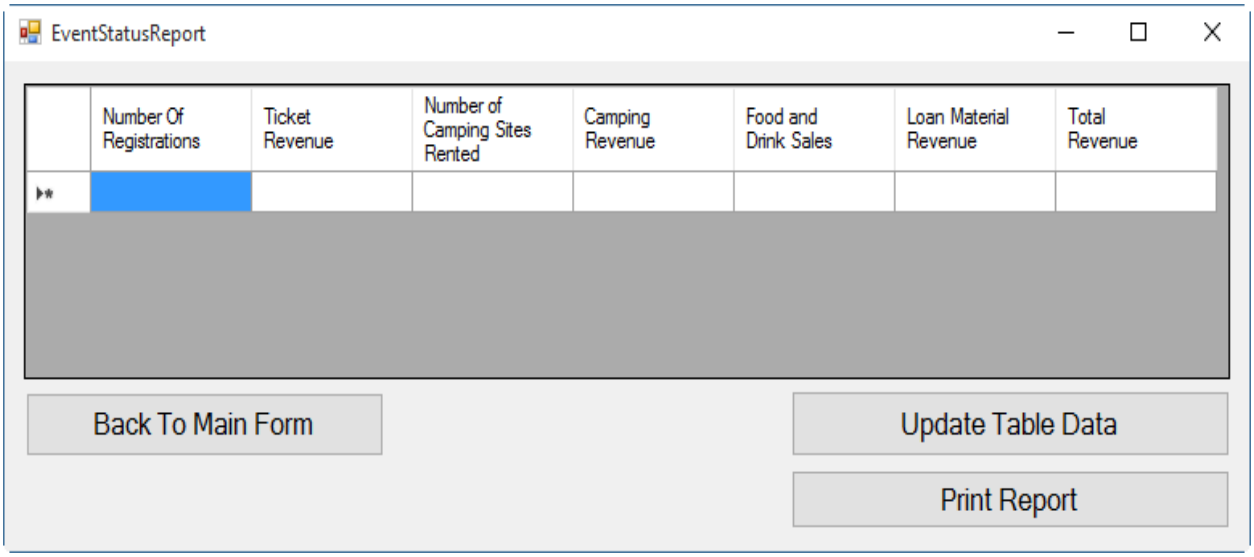
Then the staff will click on the check in button for that person and that person would have been entered into the database system as checked in (binary field). The purpose of the check in is to see if the person overstays at the camp the event organizers can charge that person extra accordingly.

7. PAY-PAL TEXT FILE APP



The pay-pal text file converting app is a very simple general app that updates the database based on information received in the text files provided by pay-pal. The text-files will be provided by Pay-pal and the event staff have to simply load and save the file to the database. Based on the information provided in the text files the financial records of the appropriate persons will be updated.

8. EVENT STATUS REPORT APP

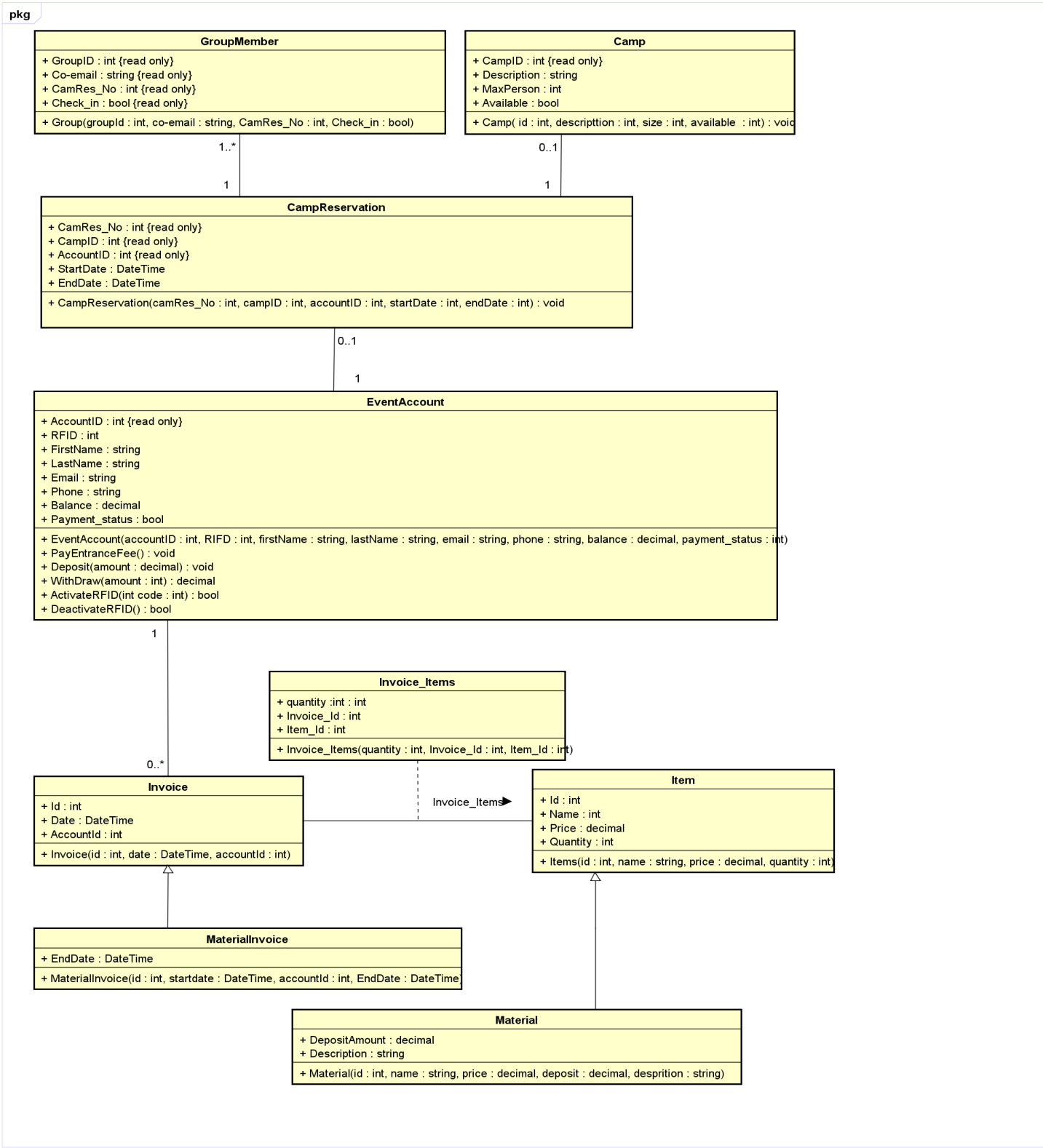


The event status report app also uses a data grid view to give an overview of key performance indicators for the event. Pressing the update table data button would provide fresh figures in the data grid view in real time. Thus the organizers will be able to see for instance, how much sales they have made with food and drinks or how much revenue they have gathered from the loaning material stalls.

This app will additionally provide a print button that will allow the event organizers to print documents of various performance information about the event. This app can be modified at later stages to incorporate any other information the client would readily like to see during the event as well.

CLASS DIAGRAM

CLASS DIAGRAM FOR BASIC OBJECTS



Event account Class

Methods:

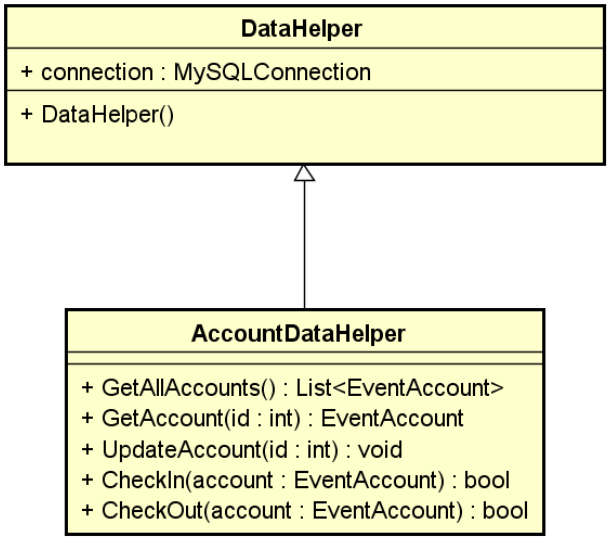
PayEntraceFee: The implementation of this method will subtract the balance by 50 euros for the entrance fee in case of the participant did not pay in advance on website (payment_status is false). The status need to be updated after payment is done.

Deposit/Withdraw: the account's balance will be added/subtracted balance by the given amount. In case of Withdraw method, the amount of withdraw must be less than or equal the balance amount. If so, the withdrawal is done and return true, otherwise this method will return false.

ActivateRFID: sets the RFID_Code by the given code in case of the RFID's value is -1 (it means that the value in database is NULL) and return true. Otherwise the method will be return false. This methods will be called in the check-in procedure of event.

DeactivateRFID: sets the RFID_Code return -1 (meaning that the client left)

CLASS DIAGRAM FOR CONNECTING WITH DATABASE AND WORK ON THE LIST OF OBJECTS



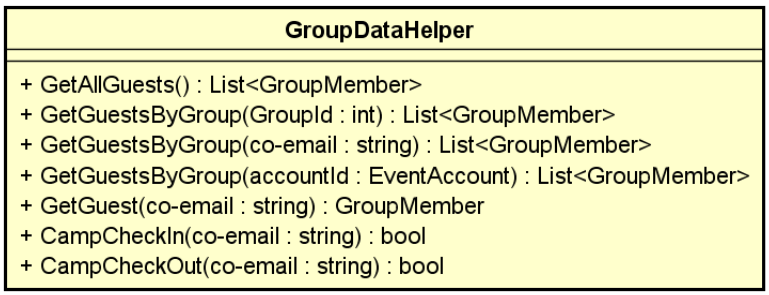
DataHelper class is responsible for establishing the connection to the database. DataHelper is the base class for all of AccountDataHelper, GroupDataHelper, CampDataHelper, CampResDataHelper, ItemDataHelper, InvoiceDataHelper, Item_InvoiceDataHelper. These classes have a role to work with database, getting or modifying the table corresponding to the table in database.

Class AccountDataHelper: contains methods relating to the EventAccount class as well as E_Account table in the database. This class should have following methods:

- **GetAllAccounts:** this method select all event account information from the E_Account table and return it under the type of List<EventAccount>

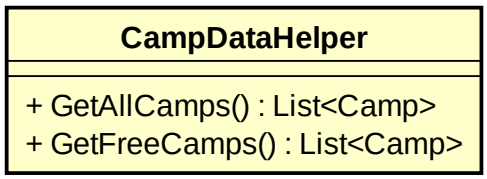
- **GetAccount:** this method should go through the list of all event accounts and return the event account base on their account id, which is provided in parameter list.
- **UpdateAccount**: this method update the event account which given account Id. The method GetAccount can be used for the implementation.
- **CheckInAccount**: this method checks find the account need to check in base on their account ID, active their RFID code. This method can be implement in 3 steps:
    - First, finding out the account need to check in – use the GetAccount method.
    - Second, activating the RFID of this account, use the ActivateRFID of EventAccount class.
    - Finally, updating this account with the changing in RFID code in to the database.
- **CheckOutAccount**: the implementation of this methods is similar with CheckInAccount. However, the method DeActiveRFID should be called.

Class GroupDataHelper: contains methods relating to the GroupMember class as well as Group table in the database. This class should have following methods:



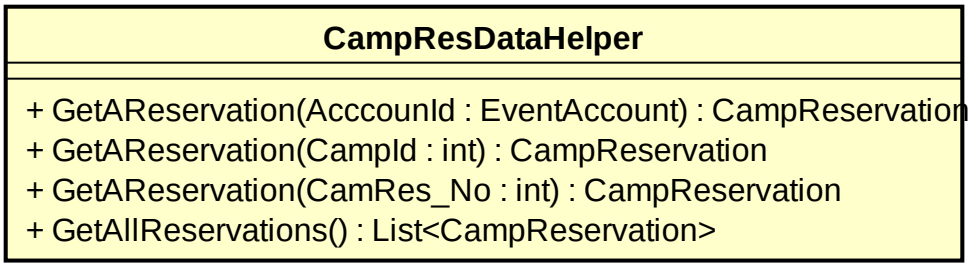
- **GetAllGuests**: this method selects all guest's information who register for camp from the GROUP table and return it under the type of List<GroupMember>
- **GetGuestsByGroup**: this method selects all of guest's information by group base on their group ID, or base on one of an email of group's member or base on the account Id of person who made the camp reservation on website (**overloading** method is used in this case) and return the list of members per group under the type of List<GroupMember>
- **GetGuest**: this method selects information of one guest base on their email and return a GroupMember object.
- **CampCheckIn**: this method will be find the participant base on their email and allow check-in into the camping area. This methods will be implement into 2 step:
    - First, the guest will be found base on their email use the GetGuest methods.
    - Second, change the Check-in properties of this object become true (if the current value is false) and update this change to the database.
- **CampCheckOut**: the implementation of this method is similar to CampCheckIn. However in the second step, the Check-in properties values need to be changed to false (in term of current value is false).

Class CampDataHelper: contains methods relating to the Camp class as well as Camp table in the database. This class should have following methods:



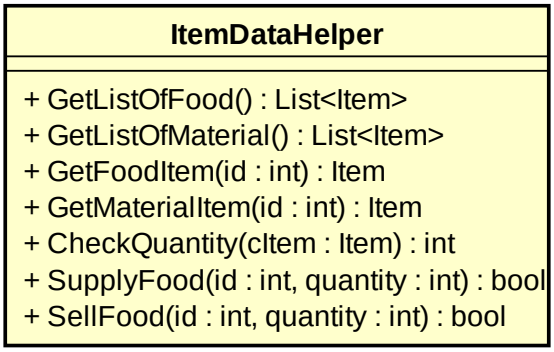
- **GetAllCamps**: this method selects all records from camp table and bind it into the **list of Camp.**
- **GetFreeCamps**: this method selects all of camps that is available for rent and return it in to the list of Camp

Class **CamResDataHelper**: contains methods relating to the CanpReservation class as well as Camping_Res table in the database. This class should have following methods:



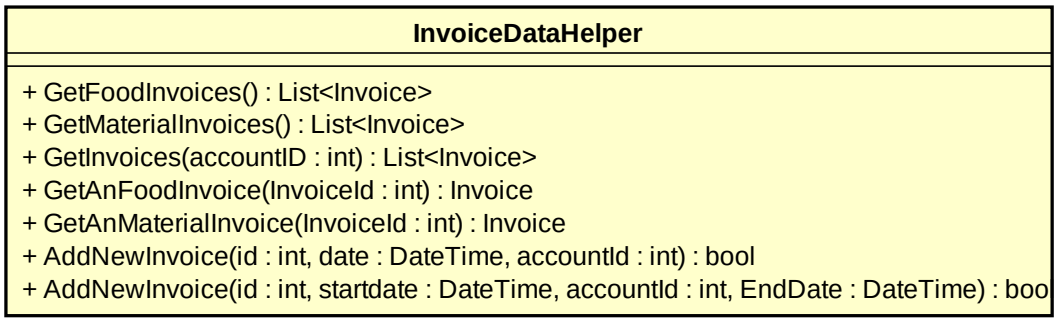
- **GetAllReservations**: this method selects all reservation for the event and adds them to the list of CampReservation
- **GetAReservation**: this method selects a reservation base on account Id of person who made registration via website, or base on one of id of the camp, or the number of camping reservation (**overloading** method is used in this case) and return a CampReservation

Class **ItemDataHelper**: contains methods relating to the Item class. Item class is the base class for Meterial class. For this reason, this class is work with 2 tables, which are Food and Material.



- **GetListOfFood**: this method selects all records from Food table in database and adds them to the list of Item.
- **GetListOfMaterial:** this method selects all records from Material table in database and adds them to the list of Item.
- **GetFoodItem**: this method should selects the record from Food table, which has the Food ID is equal to the given parameter id.
- **GetMaterialItem**: this method should selects the record from Material table, which has the Food ID is equal to the given parameter id.
- **CheckQuantity**: this method return the quantity of the given item in parameter list. If the given item is Food, the table Food will be used to get the database. Otherwise, the table Material in the database will be used.
- **SupplyFood**: the implementation of this method updates quantity of the given food id. This method can be implement in 2 steps:
    - First, the food with the given id should be found on the list of food. GetListOfFood and GetFoodItem should be used. If the id is not valid, this method return false.
    - Second, in term of the food can be found, adding the quantity in the parameter list to the current quantity of this object and return true.
- **SellFood**: the implementation of this method updates quantity of the given food id. This method can be implement in 2 steps:
    - First, the food with the given id should be found on the list of food. GetListOfFood and GetFoodItem should be used. If the id is not valid, this method return false.
    - Second, check if the current quantity is larger than the one provided by users. If not, return false. Otherwise, subtracting the quantity in the parameter list to the current quantity of this object and return true.

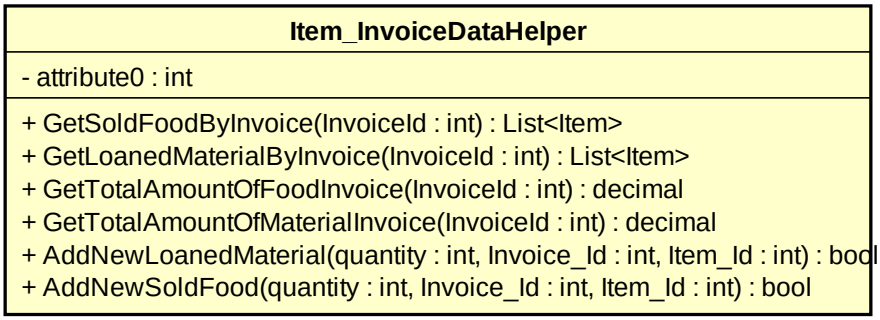
Class **InvoiceDataHelper**: contains methods relating to the Invoice class. Invoice class is the base class for MaterialInvoice class. For this reason, this class is work with 2 tables, which are FoodInvoice and MaterialInvoice.



- **GetFoodInvoices**: this method selects all records from FoodInvoice table in database and adds them to the list of Invoice.

- **GetMaterialInvoices:** this method selects all records from MaterialInvoice table in database and adds them to the list of Invoice.
- **GetInvoices**: this method should selects the invoice from both FoodInvoice and MaterialInvoice table, which has the Account ID is equal to the given account id in parameter list.
- **GetAnFoodInvoice**: this method should selects the invoice from Food table, which has the material invoice id is equal to the given material invoice' id in parameter list.
- **GetAnMaterialInvoice**: this method should selects the invoice from Material table, which has the material invoice id is equal to the given material invoice' id in parameter list.
- **AddNewInvoice**: this method should be used to add an invoice for selling for or loaning material. The list of parameter is different for two kinds of invoice (**overloading** method is used in this case). The one for invoice of food updates the FoodInvoice table, whereas the one for invoice of material updates the MaterialInvoice table.

Class Item_InvoiceDataHelper: contains methods relating to the association class – Item_Invoice. This class should have some following methods.



- **GetSoldFoodByInvoice**: this method selects all records from association **F_Invoice** table in database, which is sold for the given InvoiceID in the parameter list, and adds them to the list of Item.
- **GetLoanedMaterialByInvoice:** this method selects all records from association **M_Invoice** table in database, which is sold for the given InvoiceID in the parameter list, and adds them to the list of Item.
- **GetTotalAmountOfFoodInvoice**: this method should return the amount of money customers need to pay in the current invoice (which is given in the parameter list). The amount is calculated by the summary of the price*quantity for each food or drink.
- **GetTotalAmountOfMaterialInvoice**: this method should return the amount of money customers need to pay in the current invoice (which is given in the parameter list). The amount is calculated by the summary of the (price + deposit amount)*quantity for each material.
- **AddNewLoanedMaterial**: this method should be used to create an object of Item_Invoice and add this one to the table M_Invoice in database.
- **AddNewSoldFoodl**: this method should be used to create an object of Item_Invoice and add this one to the table F_Invoice in database

CHAPTER 4

DATABASE DEISGN

ERD MODEL FOR DATABASE DESIGN CREATION OF TABLES

In this chapter, it is contained all of informations and setup document relating to the database

ERD model:

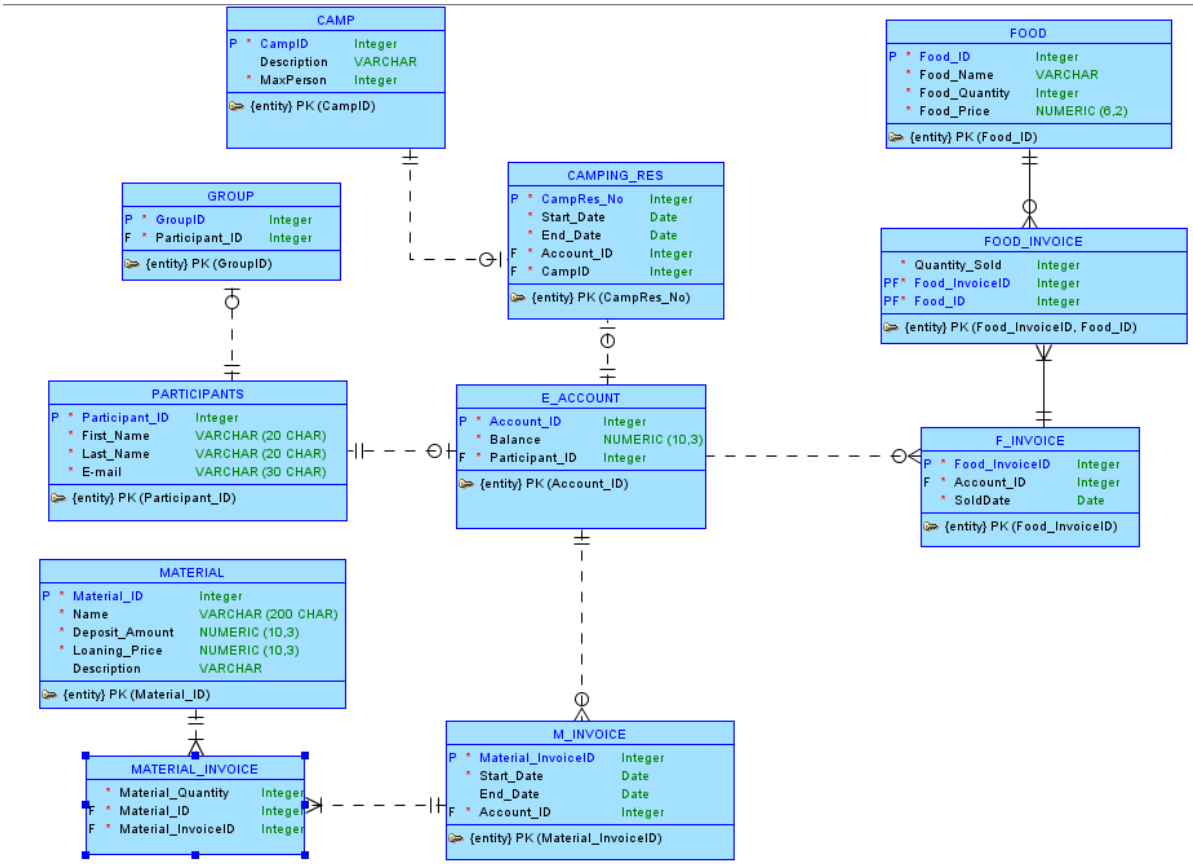


Figure 1. ERD design of database, version 1

Updating the database – Version 2

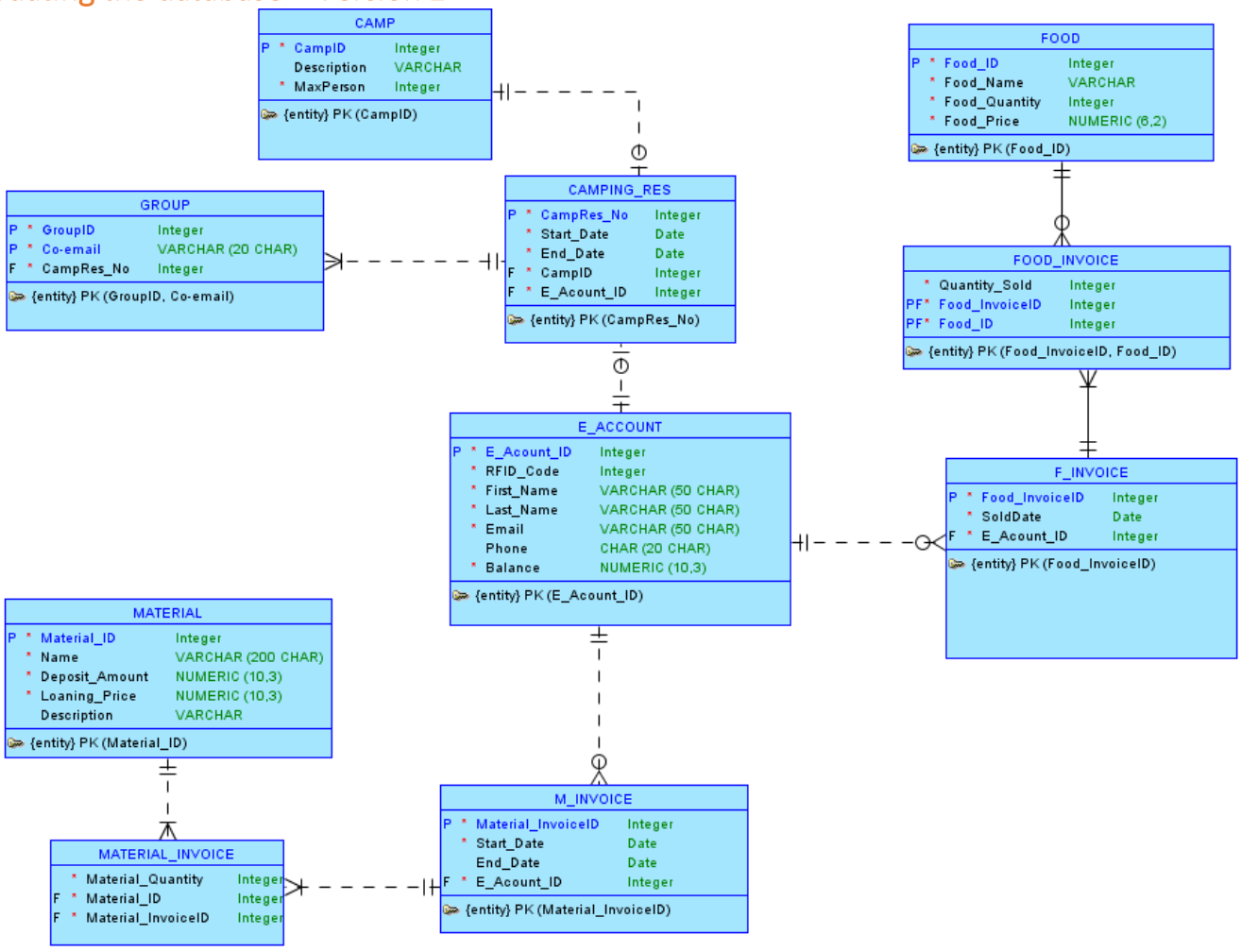


Figure 2. ERD design of database, version 2

ERD documentation:

The Database design for this event, as is apparent from the ERD diagram in figure 1 and 2, is the Event Account table/entity. In the timeline for the processes of the event an instance of an event-account is created whenever a visitor registers on the website. This registration, regardless of whether the person pays then or later, creates an instance with a unique surrogate key called event-account id as a unique identifier for that person. Everything the person then does from thereon is inherently tied to his/her event-account.

One very important thing here to keep in mind is that the RFID field for all the visitors will be null until the day of the festival when they come to the festival entrance and are issued the unique RFID. From then on that unique RFID will be attached to their own event-account.

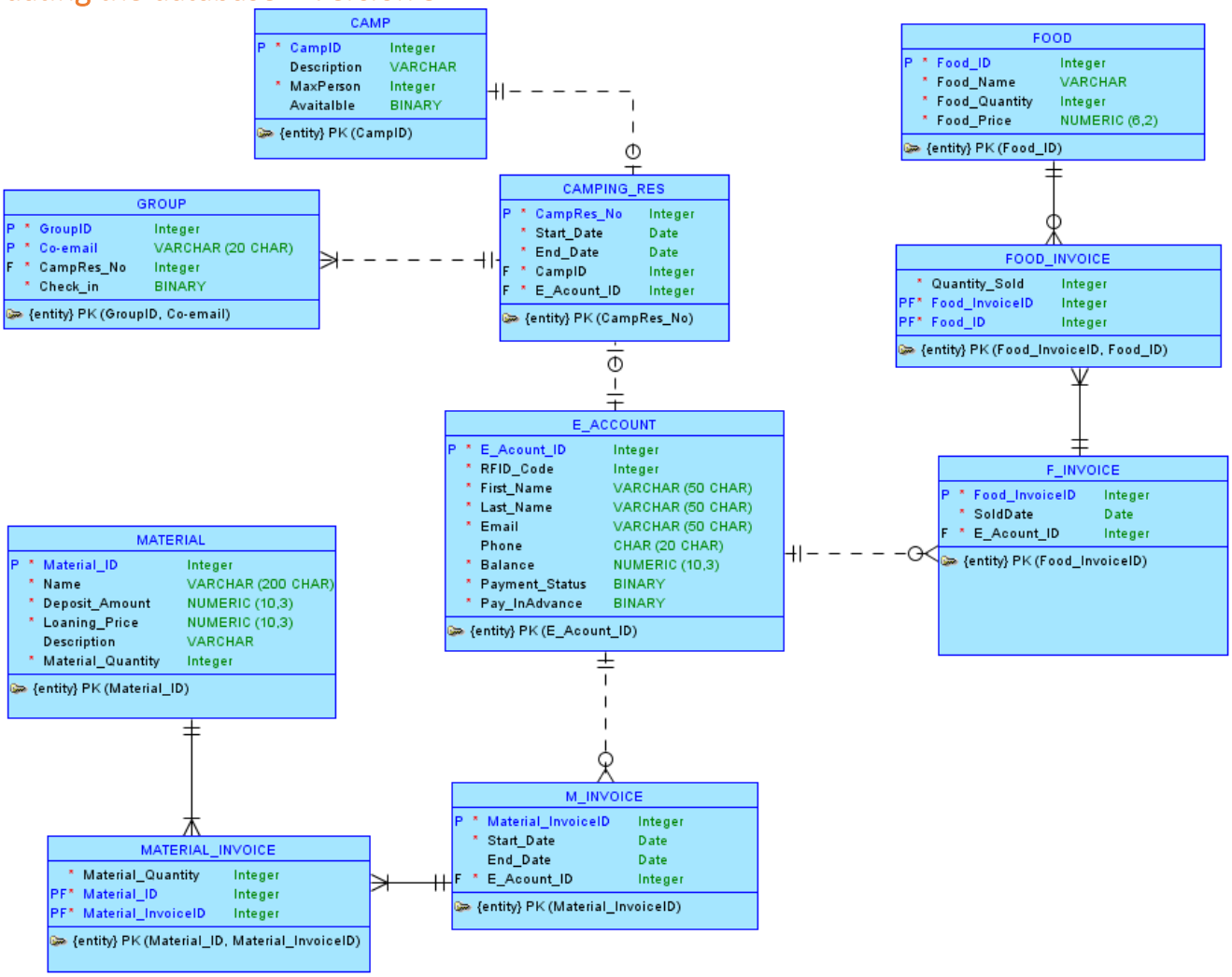
Furthermore if we were to divide the database design along the lines of the various processes that the data model serves we can split the database design into 3 main components: renting a camp, buying food and drinks and finally borrowing materials.

The database incorporates the first process, renting a camp, by having an intermediate entity called the camping reservation. The purpose of this entity is to relate a unique camping location, which is detailed in the Camp entity, to a person or set of persons who has made the reservation and secondly to relate the group of other visitors who will be joining that particular camp reservation. The benefits of this kind of construction for the camp reservation in the database would allow visitors to come to the camp site and check in to their camps even if they are without their sponsor, the person who made the reservation and paid for the camp.

The second component of the database also uses a similar intermediate table to record the food and drink purchases that the visitors make at the festival. This part of the database design also makes use of an intermediate table called F_Invoice. The purpose of the relationship between event-account and the f_invoice entity is to describe which visitor has bought what food or drink item. The f_invoice then uses an intersection table between itself and the table Food. The relationship between Food and f_invoice is that the intersection table food_invoice describes how much of a certain food or drink item were sold with regards to the f_invoice. This chain links the food and drink items sold to each visitor. The food table itself acts as an information table that contains information about the inventory of each food and drink item.

The last component of the database design is the borrowing/loaning materials during the event. This component is devised on a similar construction as the food and drink component. The only difference in this case is that there is additional information about the start and end date in the m_invoice entity. The m_invoice entity, similar to the f_invoice entity, has an intersection table between itself and the material table. The intersection table is used in this case as can borrow more than one material at a time. One can identify which person has taken what material by viewing the chain of relation between event-account, m_invoice, material_invoice and material.

Updating the database – Version 3



In the new version of database design, there are some changes:

The Payment_Status attribute is added to the entity E_ACCOUNT, which help to keep track the status of participant's payment. If the value is 1 – it means that payment is done in advance or they have not paid the entrance fee yet.

The Pay_InAdvance attribute is added to entity E_ACCOUNT, showing which participants pay in advance on the website. From the number of participants pay in advance, the total amount of entrance fee will be calculated (payment in advance will be gotten 10 euros discount). The value of this attribute is 1 if participant pay in advance.

The Available attribute is added to the entity CAMP, keeping data about which camp is rented or free. If the camp is rented – the value of this one is 1.