**Rendez-Vouz**Shmuel Malikov, Zaccharie Attias, Elyasaf Sinvani, Sharon Angado.

**I – Introduction**

Rendez-Vous is an application whose aim is to facilitate the organisation of parties, whatever their type. A registered and logged-in user, who will be called **Registered User**, will be able to create an event in which he will be both **Creator** and **Participant**. The people invited by this user will be the **Participants**. Various possibilities will be put in place in order to organize and personalize this event such as taking part in polls, making shopping proposals, finding the lowest price of the shopping list and more.

The main purpose of the application is to organize an event that will consider the opinion of each participant without headaches and endless conversations. Everyone will have their say and the creator will be able to adapt the event according to their requests.

Moreover, it is not obligatory to create an account to use the application, an option will allow a Person, provided with the code of an event in which it is invited, to participate in this last. We will call this type of user **Guest.**

It will also be possible to pay directly, from the application, its share of the total amount of a basket created by the participants via different online payment methods. This will also avoid the creator having to run after the participants to tell them how much they have to pay.

**II – System And Using Explanation**

The system works like this:

First of all, the user will have to choose between different login options: login to their account (if they have one already created), create an account by providing some basic personal information that we are used to seeing, or continue as a Guest in the application. A Registered User will have access to all the features of the application while a Guest will be limited to participating in one event.

A Guest will always have the possibility to create an account at any moment, in the application main screen, and thus have access to all functions.

The second essential action for a Guest is to enter his name and the code of the event in which he participates.

Once the user is logged in, he will have access to the application main screen. In this page, the list of events in which he participates will be displayed, accompanied by a brief summary (a guest will just see his event).

Clicking on an event will take the user to the main page of the event. All the essential information of the event will appear on a scrolling screen, in order to show the details of the event, offer to the Participant an interactive way to influence the event and stay updated about the future of the event. The Participant will see the status of the event, its date and place defined. He will also be able to see the available polls, the percentage of responses and by clicking on a survey, he will be able to select the choices that interest him. There is also the list of purchases planned for the event. He can add suggestions as he wishes. The Participant will be able to pay its share of the total amount by selecting one of the three online payment methods.

The payment system works in a simple way, the participant is given three options:

- Direct payment by credit card - entering the credit card information and saving it in the system.

- Use of the BIT or PAYBOX application - which constitute full integration with the system, when the payment is made in these applications, the money is transferred to the application system subject to the terms of use of these external applications.

In the application main screen, a Registered User may also select the option to create an event. The essential information to an event will be entered by the create such as the name, the date, the localization. The creator will have the option to choose the event participants (up to 50 participants) by selecting persons from his phone contacts. He will have the option of choosing an exact budget for each participant, so that he can present the cost of the event to each participant, and also know what the budget for the shopping cart is. And he will also be able to select if he wants to create a public and collaborative music playlist with the help of SPOTIFY.

By adding participants by creating a meeting, an SMS will be sent to the selected contacts which contains the code of the meeting and also a link to the application in order to redirect the person to the application.

A creator of an event can manage it by selecting it in the application main screen in the same way as if he wanted to participate in. He can change the name, date and localization, manage the participants (adding or removing by selected an option), manage the shopping list, create surveys and manage them (delete survey or modify information), write or edit the purpose of the event, edit the amount to pay per person, to choose the cheapest offer for the event shopping list, and to complete the event details. In the manage shopping list option, the creator can add or remove products from the actual list but he will also see the suggestions of the participants and will be able to refute or accept them in order to add them in the list. In order to create a survey, the creator will have to enter the subject/question of the surveys and enters the different possibilities of responses. In the complete event option, a SMS, which contain a resume of the event details, will be send to the participants.

In the application main screen, a Registered User can access to his profile information. He will be able to view the history of the sessions he participated. By selecting an event in the history list, it will be shown a resume of the principal information of it. An option to recover information of the event will available, its purpose is to create an event from information selected by the Registered User.

There are several more functionalities on the application, such like edit profile, delete his account, share the application and a few more, but they are basic functionality and they work the simplest way possible.

**III – Content**

The architecture of our app relies on three main things that are based on object-oriented programming.

1. User profile  
2. Particpate in the event  
3. Manage event

Editing a profile is basically the personal area of the user, where he can add a means of payment, change a profile picture, send an external link to an invitation of his friends and also change personal details.

Attending the event is the core of the app, as the username can participate in surveys, pay, keep up to date on decisions made and of course **be an important part of organizing the event**.

Finally we will talk about the possibility of managing the event, as it is an important part of the way of making decisions in the app.

As the creator of the event, I want to make management as efficient as possible, so the app gives me a **significant toolbox** like **adding surveys** that will decide important issues, **increase the budget** if necessary, **change the shopping cart** and get suggestions for more groceries, every action will be announced in the group message board To keep all participants up to date.

Once I have completed the process of building the session, event management allows me to **choose a cheap shopping cart** for us and also send summary messages to the participants.

**IV – UML’s**

During the process we created a number of diagrams which helped us to see the processes which are carried out in the app more clearly, it should be said in an abstract way.

The diagrams focus on a number of topics, such as: objects, actions, and processes.

* Class Diagram:  
  There are five departments in our system, such as: location, survey, meeting, user and person.  
    
  The **location** class is a direct contact with a meeting class, as it provides the user with the time of creation of the meeting and also with its completion determining the location of the future event and thus updating the participants in real time. The function in the class allows you to create a string that contains all the fields of the object that contain the country, city and address of the event and provide it to the meeting class.  
    
  The **Poll** class maintains a two-way integration with the meeting class, where it allows the event creator to conduct surveys, change the answers and get the answer for each survey.  
    
  The **Meeting** class is a class full of important operations because it is a mainstay of the system.  
  The class contains fields such as: determining the name of the meeting, date and time, location and adding participants, creating a collaborative playlist on Spotify (by fully integrating with it), surveys that will influence important decisions in the nature of the event and the possibility to pay.  
    
  **Person** class is a user's base class, it describes an option where an app user will want to log in as a guest so we will only require his name and number and be given basic options in the app.  
    
  And finally the **User** class, contains fields that are the user's details, such as: email, password, date of birth, profile picture and meeting history.   
  Using the object's methods you can change the values of the user's personal profile.
* Block diagram:  
  A block diagram represents processes in a black box, which we do not specify the exact processes that happen in the application, we examine the process by way of input and output.  
    
  For example, by creating a event - the user will enter input into the system to create a session, this input will be used for managing the event interface and presenting the event details to the participants as an output.
* Use-case diagram:  
  Using this diagram we will describe the functionality of our application by clarifying how an action entails another action.  
    
  **Sign up** allow non user to input there details to the system and create valid user that can use the full options of the application.  
    
  **Login as a guest** allows the non-user to reach the lite version of the app, which has limited options of actions, such as: responding to a survey, adding an offer to a shopping cart and paying for an event.  
    
  **Login as a user** allow registered user to enter the application by providing email and password, or sign in via google or apple-id.  
    
  **Share the app** allows the user to send a link to an event outside the app, via WhatsApp, messages and more.  
    
  **Seeing the event history** allows a registered user to view events that have already been completed, to renew specific details from them in favor of a new event.  
    
  **Edit profile** allows the user to edit the fields of his user profile, such as photo, email password and more.  
    
  **Create new event** allows the user to create a new event, create content for the event and add participants by invitation in the message.  
    
  **Manage event** is an action that allows the event creator to add surveys, change event details, add items to a shopping list and receive offers from attendees, change the budget set for the event and also complete it and get the cheapest offer for the selected shopping cart.
* Activity diagram:  
  An activity diagram describes navigation in the application, showing how every action that the user does leads to a certain page or another action.  
    
  A registered user can return to certain options at any time, an example of these options:  
    
  1. Choose option from main screen  
  2. Edit profile  
  3. Share the application  
  4. Delete account  
  5. See the history of the events  
  6. Create an new event  
  7. Manage the event  
  8. Choose event  
  9. Sign up  
    
  The green and red arrows describe the return options that can be returned in the system.  
    
  For example after watching a particular event from history, we can go back to the main screen of the app.  
  Also the same options of repetition exist in sharing the app, editing a profile, creating an event.  
    
  Deleting a user takes us back to the home screen where you can register and log in.  
    
  In event management, any action derived from it brings us back to the general page of event management.  
    
  After selecting the event, any action that the user performs can be returned to the event page and perform additional actions on it.
* State diagram:  
  A state diagram describes the behavior of an object in certain situations, from the beginning of its creation to its end.  
    
  **Meeting:**  
  The object has four modes, in the initial mode we enter information, we reach a junction of two modes:
* a mode where it can add participants (if the number of participants is less than fifty).
* the second mode the number of participants is fifty and can only remove participants.  
  These situations drain into one situation which is the payment, After the payment the object reaches two options:
* Continue change details in the object
* The lifespan of the object ends when the event completed.  
    
  **Person:**  
  The guest has three modes, the initial state in which he enters information into the system and the object is created.  
  He is now given two options:
* Creating a user in the system, if he selects this option, the lifespan of the guest object ends and a user-type object is created.
* Participation in an existing event to which he has been invited, the guest enters information and during the participation information is entered by him.  
    
  **User:**  
  The user has three modes, the initial state in which he is a guest is registered and as a result a new user-type object is created whose profile can now be edited.  
  He is now given two options:
* By clicking the Delete profile option, the object is deleted from the database and its lifespan ends.
* Change information to change the user's profile information.  
    
  **Poll:**  
  The user has three modes, the initial mode in which he enters information and a survey type object is created.  
  He is now given two options for situations:
* Create another survey-type object and change its details.
* Deleting an existing survey thus ending the lifespan of the object.  
    
  **Location:**  
  The user has three modes, the initial state in which he enters information and a position-type object is created.  
  He is now given two options for situations:
* The meeting is still open and in process, so it is possible to change the details of the object.
* The encounter is over, so the lifespan of the object ends.
* Sequence Diagram:  
  The sequence diagram shows process interactions arranged in a particular time sequence. It describes the processes and messages that are transmitted during the exchange between the two.  
    
  Note:   
  Because we went through the registration and login process, we chose not to go into detail about them, as these are obvious processes and also, we emphasized their functionality in other diagrams during the assignment.  
    
  **Main flow charts to describe:**1. See event main screen to participated  
  2. See a specific event in the history   
  3. Create event  
  4. Manage the event  
    
  When watching a particular event, I click on it, then pull information from the repository and then from a meeting object, the information is transferred back to the system and printed on the user's screen.  
    
  When you select the event history option, the diagram splits into two modes:  
  I click on it and ask the database to show me the events that have already taken place, from there it accesses a user-type object, and it shows me the entire event history of the user, the events are printed on the screen.  
  If there are no events, an appropriate message will be displayed stating that there are no events in the database.  
    
  When selecting the option to create a meeting, the user must first enter information, then a meeting type object is created that is stored in the database, the database returns the object details and the meeting is printed on the screen to the user.  
    
  When choosing the option to conduct a meeting, all sub-diagrams work on the same principle. I access the system and demand to change information in the repository, then access the meeting object and update it and finally the object is updated in the repository and passes it to the system, finally output is displayed to the user that the information has been updated successfully.