**She codes;**

**Design document**

Project: Event planning

Student: Noy Gini

**Introduction**

**Scop**

This web application will be used for planning events such as birthday parties, bachelorette parties, or any other event that requires prior planning. The app is designed for the planners of the event and will provide a comfortable way for multiple planners to monitor all the necessary things for a successful event.

**Overview**

In this document I will start by explaining about the different uses and features of the application. I will describe in detail the apps design and how it integrates with other systems. Last, I will show how the app will be tested and ready for the users to start enjoying it!

**Terminology**

Planner – user of the app who has an event planed.  
Invites – people who are invited to the planners' event.  
Attending – Invites that confirmed arrival to the planners' event.

**Software design description**

**General flow**

When first opening the app, a list of all events that the user was/is planning is displayed. By pressing one of the events, the user will access a page which includes 5 options: invites list, attending list, refreshments list, equipment list and expenses list.

Invites list will include all the people that were invited to the event. Next to each name, there will be an option for the user to mark the person as coming, and his name will automatically be passed to the attending list. The refreshments list will be a table with two columns – one for the food, and the other for the person who is responsible to bringing it. The equipment list will contain 3 tables, one for equipment needs to be bought, other for equipment that was bought, and another for equipment someone brings from home. Like the refreshments table, the second column in each table will include who is responsible for buying/bringing each item. The user could drag items from one table to another. The expenses list will allow the planners to write which planner paid for what, and then the total sum for each planner, and how much money the other planners owe him (or vice versa) will be displayed at the end of the page.

Data structure

Login

Sign Up

**My app**

 Creating account for new users   
(maybe with Gmail or Facebook)

Authentication

For an existing user

Event list

|  |
| --- |
|  |
|  |
|  |

Invites list

|  |
| --- |
|  |
|  |
| תיבת סימון עם v עם מילוי מלא |

Attending list

|  |
| --- |
|  |
|  |
|  |

Refreshments

|  |  |
| --- | --- |
| brings | food |
|  |  |
|  |  |

 Equipment

Bought:

|  |  |
| --- | --- |
| responsible | item |
|  |  |
|  |  |

Brought from home:

|  |  |
| --- | --- |
| responsible | item |
|  |  |
|  |  |

needs to buy:

|  |  |
| --- | --- |
| responsible | item |
|  |  |
|  |  |

1

 Expenses

|  |  |  |
| --- | --- | --- |
| Who paid | cost | item |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| Total cost paid | planner |
|  |  |
|  |  |

Planner 1 owns x$ to planner 2

Main functions:

* Sign up / Login user /Log out
* Event list: add event, delete event…
* Add other user as planner to an event
* Invite list: add a guest, remove guest, mark guest as attending
* Attending list: add/remove guest
* Refreshments: add/remove item and the person responsible to it
* Equipment: add/remove item and the person responsible to it (function for each table)
* Expenses: add/remove item, its cost and the person that paid,   
  calculate total cost for each planner, calculate balances

Software architecture

// not finished

I will use the next classes:

* Invites class
* Attending class
* Refreshments class
* Equipment class
* Expenses class
* Event class – contains details about the event and one object from each class described above.
* Events class – includes a list in which each element is "event" type

User interface

* Sign up, login, log out
* Create/delete an event
* Invite other user to co-plan an event
* Maintain invites list with add and remove buttons
* Mark invites as coming (will copy it to the attending list)
* Maintain refreshments list with add and remove buttons
* Maintain equipment list with add and remove buttons,  
  option to drag an item from one table to another
* Monitor expenses with add and remove buttons

Data handling  
I will use sqlite3 database for the application. I chose this database because this is the database that comes with Django so for my first app it makes it easy to use, and also as I read, SQLite is a [small](https://www.sqlite.org/footprint.html), [fast](https://www.sqlite.org/fasterthanfs.html) and [high-reliability](https://www.sqlite.org/hirely.html) DB engine, most used in the world, and therefore fulfills my apps' requirements. I will save data about the users in the system with their identifying information. I will also save the events, and for every event the users that plan this event, and all the needed information in the tables described at the software architecture section.

Integration with other systems

The software will interact with other systems such as systems to register a user to the app (maybe with Google or Facebook accounts). Also, it will require direct access to the databases because most of the information in them is exposed to the user. For communication protocol we will use HTTP.

**Tests and Monitor**

Tests  
To check that my app works properly, I will start by running the following tests:

* Sign up and check that the user is added to the database
* Login / log out and check that it works correctly
* Create an event and check that it appears in the database with the right details (name of the event, name of the planner...)
* Delete an event and check that it is no longer in the database.
* Create an event and add another user as a planner. Check that it appears correctly in the database. Then, remove the planner and check the database, and also that there is no longer access for this planner to the event
* Add a name to the invites list. Check if it appears in the database. Then, erase it and verify that after deletion it is no longer exist.
* Add a name to the invites list. Mark near that name that this person is coming, and check if a copy is created for the attending list.
* Add a name to the attendings list. Check if it appears in the database. Then, erase it and verify that after deletion it is no longer exist.
* Add an item + name to the refreshments list. Check if it appears in the database. Then, erase it and verify that after deletion it is no longer exist.
* Add an item + name to each one of the tables in the equipment's list. Check if it appears in the database. Then, erase it and verify that after deletion it is no longer exist.
* Add an item + cost + name to the expenses list. Check if it appears in the database. Then, erase it and verify that after deletion it is no longer exist.
* Add a few rows of item + cost + name to the expenses list. Check if the calculation of total sum to each planner is correct.

Other tests will check that there is no "wrong" behavior:

* Trying to add two users with the same username, and check that the app does not allow it.
* Adding an item to each one of the optional tables and check that the data does not appear for other users of the app.
* Trying to logout in the middle of adding the item to each one of the tables and checking that a message appears to make sure the user wanted to exit the app.
* Trying to add text with length over X chars and check that a message appears to let the user know that his text is too long.

Next, I will check that all the above are shown correctly in the app for the users (for example, when adding an item, it is displayed in the correct location for the user). I will also do a performance test – in order to see that the app can endure a lot of users at the same time. Finally, I will check that the user interfaces is as it was designed (colors, sizes, text font, etc.).

Logs

Thing that are going to be logged: User sign in the app, adds an event, joins an event of other user, add/delete an item to/from each one of the tables. This will help me to detect bugs or problems in my app and learn statistics about the use of the app (how many people use the app every day for example). Because this app allows multiple users to plan together an event, I will keep for every item that was added the name of the planner that added it, and the time it was added at.

Alerts  
alerting rules:

* User deleted data which should not be accessible to him from the database.
* The app is overloaded with users.
* The database is overloaded. There is not enough memory to save all data.

A/B Testing

I will create a few versions, each one with a little different UI – size and colors of the buttons, background colors, size and font of text, etc. I will ask users to rate the app visibility, and the UI with the higher score will be chosen. I could also use the data about users log in the app to understand which version attracts more users.

Future Develop

The app as I presented above is designated to the planners of the event. A way to extend the app is to also open it for the event participants. In this way, participants could mark they are coming to the event, see who else is coming, mark what kind of refreshment or equipment they want to bring, and so on. The expenses section will stay open only for the planners.

Summery

This application will serve every group of people that wants to plan together an event, and will offer many built-in parts for the user to organize his data in an easy and comfortable way.