SPRINT RETROSPECTIVE

Software Engineering II - Group E

Process measures

Macro statistics

	COMMITTED	DONE
Total number of stories	**24**	24
Total number of stories in sprint (from story #14 to story #22 + #25)	**10**	10
Total points in sprint	**13**	13

Detailed statistics

STORY ID	NUMBER OF TASKS	POINTS	TOT HOURS ESTIMATION	TOT HOURS SPENT	HOURS PER TASK
0	6		4	6	1
14	2	1	2	3	1,5
15	4	1	4	6	1,5
16	2	1	2	3	1,5
17	2	1	3	3	1,5
18	3	2	5	6	2
19	2	1	2	4	2
20	3	2	4	8	2,667
21	5	2	4	5	1
22	1	1	1	3	3
25	1	1	1	2	2
UNIT TESTING			8	13	
E2E TESTING			3	3	
CODE REVIEW			6	6	
TOTALS			**49**	**71**	
HOURS PER TASK		MEAN	1,788	STD DEV	0,628
TOTAL TASK ESTIMATION ERROR RATIO			0,69		

Story#0:
Install phpStorm and configure phpunit on all the pc of the group (to have the same environment for testing).
Configure composer on all the pc of the group.

Quality measures

- **UNIT TESTING**

Total hours estimated: 8h
Total hours spent: 13h
# of automated unit tests: 40
Coverage: 43,6% (weighted average between coverage of db.php and functions.php)

We decide to test only meaningful functions (in the code they are all commented).

- **E2E TESTING**

Total hours estimated: 3h
Total hours spent: 3h

- **CODE REVIEW**

Total hours estimated: 6h
Total hours spent: 6h

Assessment

1. What caused your errors in estimation (if any)?

   We did not understand deeply (when performing the estimation poker) how much certain stories were difficult. To be specific, we underestimated the difficulty of uploading and storing files on the server and being able to reach them again from another page.
   Moreover, we encountered some difficulties in managing the calendar (same function for three different reasons).

2. What lessons did you learn (both positive and negative) in this sprint?

   We improved the readability of code to make the code review easier. We also organised differently our work, in fact this time we did not assign one story per person but we split the team in two groups, one for developing (with more than one story per person) and the other for testing and support when needed.

3. Which improvement goals set in the previous retrospective were you able to achieve?

   We managed to avoid writing code that needs refactoring through better communication.

4. Which ones you were not able to achieve? Why?

   We improved the quality of the code, but we think that we can do better. The code is not perfectly commented and understandable by others.

5. Propose 1 or 2 improvement goals for the next sprint and specify how to achieve them (technical tasks, team coordination, etc.)

   Some commit made the tests fail (for semantic changes or simply because they introduced bugs), so for the next sprint we want to execute the tests BEFORE every commit, eventually adapting them if the function had changed semantically.

6. One thing you are proud of!!

   During this sprint planning, we recognised that some of the stories did not have the right amount of story points. We are proud that we were able to identify the "problematic" ones, to perform a good estimation and to complete all of them inside the time budget of the current sprint.