Project Final Report

Software Engineering In Practice – ECSE 428

Due Date: December 5th, 2021

Debray Juliette 260 850 966
Destiné Maxens 260 926 906
El Mahmoudy Riad 260 853 502
Guerre Antonin 260 859 036
Mesam Mohammad Salman 260 860 161
Tan Sean 260 761 889
Wener Dominic 260 871 052
Xu Glen 260 767 363
Yalcinoz Gorkem 260 710 053

Team Name: Full Moon **Project Name:** Full Moon Trivia Game Application

SCRUM Masters	Riad El Mahmoudy	Juliette Debray	Mohammad Salman Mesam	
SCRUM Objects and Rituals	**Sprint 1**	**Sprint 2**	**Sprint 3**	**Improvements for Sprint 4**
Done Checklist	- Some checklist tasks were prerequisites of other tasks, but the initial task distribution caused blocks and time wastes. - In the end, some members had to work on the same task together due to dependency issues, but those were not fully shown on the checklists. - Code Re	- Updates from each member were given every week - Tasks were marked In Progress or Done at each update - Members were given autonomy to complete task list under restriction of letting the SCRUM master know	- We went through the checklist every time a story was finished to mark it as done. - No more code was developed by the team for a story after its status was marked as done..	- Keep updating the done checklist as stories and tasks get completed - Add a column in the done checklist to add remarks and comments about a given task
Product Backlog	- Product backlog was discussed during the brainstorming session for the project idea. - Knowing the time constraints, minimum requirements for the project dictated the features chosen and added to the product backlog. - Product backlog does not necessarily follow a priority	- Features for the sprint were chosen at the planning session - Less features than sprint 1 to avoid leftover tasks - After retrospective meeting and reevaluation of our capabilities and feasibility of our project, new features were added and others were scraped at planning session	- Maintained an excel doc on the Google Drive that has the priority and a status of whether a particular user story is groomed or not. - For every user story that we work on we have a feature file referenced that has all the details. - Only stories that we planned for the sprints have a feature file associated with them.	- Keep the same format as the one that was used during the previous sprint - Keep adding feature files in the dedicated folder for every new created feature

	order. The priority was determined at the beginning of our grooming session. - Feature files for each feature were discussed during our grooming session for Sprint 1 and added right after - No changes were made to the product backlog during Sprint 1 - 8 Features were picked and groomed for Sprint 1 which was	- Groomed tasks were marked as such in the product backlog. - Feature files were also discussed and added around the time the features were groomed by the team. - 6 Features were picked this time, with the leftover tasks from Sprint 1.		
Sprint Backlog	- No separate sprint backlog was created. - Additional set up tasks were added as this was the first sprint. - The user stories picked for Sprint 1 were color coded as such in the product backlog. - Each user story was broken down into multiple tasks, categorized as Backend, UI, Model, Setup, and Testing.	- Leftover tasks from the previous sprint were transferred onto Sprint 2. - User stories were broken down the same way as sprint 1 and categorized the same way as the Sprint 1 features. - The features picked for Sprint 2 were color coded as such in the product backlog	- We did not maintain a separate Sprint backlog. We kept updating the priority of the stories in the product backlog with the Sprint they were associated with.	- Create an additional document dedicated to a single sprint backlog and integrate it to the product backlog at the end of the sprint
Sprint Task List	- Built by the SCRUM master - A proper format was only drawn up at the end of the sprint	- Built by the SCRUM master - Followed same template as Sprint 1 - Built right after the	- The format that was used for Sprint 2 was kept for Sprint 3 as it yielded good results - Added an uncertainty factor to the time estimation to	- Keep the same format as Sprints 2 and 3 - Add a column for an updated time estimation that could be modified right after

	- The time estimation for each task was done mainly by the SCRUM master and checked by the other team members, which led to an inaccurate time estimation - The initial tasklist focused on the set up too much and created a large amount of tasks dedicated to the set up - Many modifications were made during the sprint - A task description and acceptance criteria for each task were only added at the end of the sprint.	preparation meeting, once the features were decided upon - Added more detailed acceptance criteria at the beginning of the sprint - Added a more thorough description of each task at the beginning of this sprint - The time estimation for each task was done by its assigned developer and checked by the other team members and the SCRUM master which led to a more accurate time estimation	account for possible future improvements of a given task	a developer has started a task if he/she realizes that it is going to take more or less time
Burn Down Chart	- No burndown chart was required for sprint 1	- Burndown chart was built in terms of weeks - Weeks were deemed more accurate than days, since member updated their progress to the SCRUM master weekly (and not daily) - The y-axis represented the number of tasks left to complete - Result followed a relatively linearly decreasing curve - Unfinished tasks made it	- Burndown chart updated with number of tasks left at the end of every week of the sprint - Burndown chart visualized in a graph - No tasks were left over at the end of this sprint, which made the burn down chart closer to ideality	- Have a more detailed day-by-day burn down chart - Add another axis for the remaining effort in terms of hours

		imperfect		
Backlog Grooming	- Feature files were written at the start of the sprint but by only one member - Feature files contained alternate flows and error flows. They were fairly extensive - User stories picked were detailed and discussed by all members	- Tasks were chosen by the entire team during the preparation meeting - The feature files were written by multiple members (more than in sprint 1, but not by all) - Feature files were also added to the missing ones from sprint 1	- The team met before the start of sprint 3 and decided which stories we would groom for sprint 3 - Feature Files were drafted for the stories selected for sprint 3	- Keep the same format that was used in Sprint 3 - Have 100% team participation in the decision-making for which stories will be done during the sprint
Sprint Planning	- Too many tasks and stories were included in Sprint 1 - Did not take technology ramp up into account - Too confident in the amount of work we thought we could do - Fairly bad time estimation for the Sprint as a whole - Bad work distribution during the week led to frantic weekends - A few tasks had to be carried over to the next sprint	- Better balance than during Sprint 1 - Fewer tasks and stories were implemented but the result was better as a whole - Quality over quantity approach - Better week-long work and team organization - Very few tasks had to be carried over to the next sprint	- Even better balance and planning than for Sprint 2 - Same amount of work was undertaken as during the last sprint - Same level of successful output - Better organization and time-work distribution - No task had to be carried over to the next sprint	- Keep a reasonable number of tasks and stories in order to prioritize quality over quantity - Keep the same work distribution over the course of a week to avoid last-minute work
Story Estimation	- Too many stories were included in Sprint 1,	- Based on resulting efforts from Sprint 1 for repetitive	- Estimations were based on 2 factors:	- Keep the same format as the one that was used for Sprint

	resulting in unrealistic work amounts. - This led to us having unfinished tasks at the end of the sprint	tasks such as integration testing and service testing, story estimations became more precise. - New UI tasks were poorly explained and added to existing stories	- Experience with planning from sprint 2 - Members provided an estimate of how much time they expected based on their skillset/techstack. - Taking into account the integration of the 2 previous sprints	3 as it focused on the skills of certain developers
Task Estimation	- Assigned stories to team members rather than tasks - Poor task estimation and hard to split work evenly or ask for help - Ended up with a non-homogeneous look - Many dependencies between tasks assigned to different people created delays and unnecessary blockers	- Multiple team members worked on different tasks within the same story - Better structure allowed us to share our successes and impediments with the team - This created better problem solving and team communication between members assigned similar tasks in different stories	- Improved Sprint 2 format - Multiple members working on different tasks within the same story - Tried to alternate task types between Sprints (team member who worked mostly on the backend during Sprint 2 worked on UI during Sprint 3, for example) - Even better communication than during Sprint 2 - More back-and-forth ideas about how to improve our application as a whole	- Keep using the task estimation and distribution format developed in Sprint 2 - Now that we have seen who is more comfortable with what part of the project, we can have a more efficient task distribution - Keep communication up between members assigned similar tasks in different stories
Scrum Meetings	- 2 meetings over the entire sprint - Meetings were too long, a lot of technical aspects were discussed and a lot was necessary to catch everyone	- Switched to a having meetings mid-week: retrospective over the previous week and planning for the rest of the week - Most members were present	- Improved Sprint 2 format - One hour-long meeting with all the team members each week - Even smaller meetings with	- Keep using the Scrum Meeting format developed in Sprint 2 - Keep the weekly meetings to make sure everyone is on the same page and gets a good

	up - All members were present	(all → 6 members → 7 members) - Scrum master would then update the members who could not make it individually or in a separate meeting with all of the missing members	the scrum master - About 3-4 members showed up to each meeting about different topics - More frequent meetings than during Sprint 2 - Scrum master would update the missing team members	debrief of what has been done during the week - More small-scale meetings to promote idea sharing and communication about technical issues or ideas
Sprint Demo	- The following things were included as part of the demo: - Weekly Task List, including the details in task description and requirements. - Selected static UI pages. - A screenshot of tests passing for this sprint.	- The following things were included as part of the demo: - Weekly Task List. - UI pages of login, sign up, and account settings. - Screen recording of the working UI components - Database updating simultaneously with frontend. - All feature files - A screenshot of tests passing for this sprint and previous sprint. - Meetings notes - Burndown Chart	- The following things were included as part of the demo: - Video Recording of the End to End work that was done during the sprint. - All feature files corresponding to the user stories that members worked on - A screenshot of test passing for current sprint and passing tests from all previous sprints. - Any meeting notes that were taken down - Product backlog included as well - Burndown Chart	- The following things are planned to be included as part of the demo: - Live demo of how multi-player function works normally, and how the database is updated simultaneously with be frontend. - Burndown Chart - Video Recording of the End to End work that was done during the sprint. - All feature files corresponding to the user stories that members worked on - A screenshot of test passing for current sprint and passing tests from all previous sprints. - Any meeting notes that

				were taken down - Product backlog included as well
Sprint Retrospective	- Had a very short Sprint retrospective as we were late with some of the tasks - Decided to take on less tasks than we did for Sprint 1	- Discussed how this sprint was a lot more organized than the first one - This was due to the fact that we took on fewer stories - Only 7 members could make it to the meeting we had about the sprint retrospective, so we could not get everyone’s feedback	- This sprint retrospective was very similar to the Sprint 2 retrospective - All of the team members came to that meeting, which led to us having a more thorough feedback of everyone’s experience	- Keep the same format as the one that was used for the first three sprints: a meeting in the days following the conclusion of a sprint to talk about what went well and what did not - Include retrospective on entire project, as this should be the final sprint
Team Collaboration	- Poor team collaboration in terms of talking about tasks and the way of implementing them - This led us to having very heterogeneous code at the end of the sprint - Very good communication in terms of technology ramp-up as we helped one another get on the same page about the technology and frameworks we would use for this project	- More regular communication (meetings and text updates) aided communication and collaboration on coding-related aspects of the project - The scrum master would make sure everyone (even the team members who could not attend the meetings) was on the same page about where the team was on the project	- Scrum Master facilitated collaboration between members by setting up all meetings and coordinating availabilities of members - Scrum Master helped in removing any blockers faced by members on the team by facilitating communication between members. - Use of Facebook Messenger and Zoom was made for all communications.	- Keep the same communication format as the one that was used during Sprint 3 - Try to either have more meetings with few team members to promote communication or to have a bit fewer meetings but try to have as many team members attend at a time
Continuous Integration	- Setting up basic gradle and Travis CI, making sure the	- No change to Continuous Integration in this sprint.	- Automated the testing process to be triggered using	- Keep the automated build and testing functionality

	build triggered and Travis ran the build using Gradle.		Travis CI for every push to the master.	- Split up the controller and service automated testing to get a better idea of what is passing and what is failing
Acceptance Test Automation	- No automation yet, the tests were run manually.	- No automation yet, the tests were run manually.	- All tests ran automatically at every build.	- As indicated above, split up the controller and service automated testing to get a better idea of where to check if the automation fails