Sprint #01 Retro

Date: 02/09/2020

Sprint: 1

Scrum Master: Vincent Villaflores (s3728807)

Development team: Meng Kheang Leng (s3704080), Chhayhy Kourn (s3699618), Hue Phuong Le (s3687477), William Bossen (s3658961)

Things That Went Well

Throughout this sprint, all team members are still able to meet twice a week on an agreed time to update on how we go with our tasks and get additional assistance from other team members immediately if any difficulties arise. The workload has been fairly well managed and coordinated, everyone seems to be efficiently working together with increased agility.

The basic front-end components are already implemented (home page, login page, user register page, add new employee form page, create a new booking form page, view current booking page, booking history page, available worker page, view employee page). It will give us an overview of how our front-end components are connected together, adding functionality to them will be less challenging in the future.

Implementing the back-end is challenging at the beginning as we are all new to Spring Boot. However, it still went smoothly and all the controllers, repositories and services are successfully implemented.

Things That Could Have Gone Better

As team members have already sorted out what to include for this sprint by the end of the previous sprint, there were less confusions to be cleared out before the start of this sprint. However, the “create new booking” features required more work than we anticipated. To ensure we completed the sprint on time, we needed to allocate more time towards the tasks necessary for those features.

Although some members of the team finished the SEF course before, some are still new to the version control system. This is due to us coming to the realisation that we should perform more commits to Github frequently.

Things That Surprised Us

Working with date and time, arranging the new booking duration data in the back-end is more difficult and less straightforward than we expected. When it comes to start implementation, the process for converting the Date object to the correct format and save to the database takes much more effort. The development team also has to figure it out how to display the valid time that a customer can make a booking to prevent more unexpected errors later on.

Additionally, we decided not to implement a logged-in function for this sprint, so our front-end developers are struggling to distinguish between different types of users to display different dashboard.

As mentioned in the last sprint, all team members are also new to both Spring boot and React, resources to understand this concept were limited during the beginning phase of the project. Therefore, it takes us much time to search for tutorials and learn how to pull data from back end to front-end. Also learning how to write unit testing is much more challenging as it is totally the new concept (Mockito to mock for services and controllers, also dealing with the mapping relations when testing repositories).

Lessons Learned

Based on our experiences in this sprint, potentially we need to invest more time in the implementation of the different account types, then it will be less struggling for any other features.

There are lessons that can be learned. For instance, to always expect for the unexpected. As we were quite confident that the unit testings are less important and easier compared to actual implementations of back-end and front-end. The team members thought we could postpone the writing of unit tests and focus on coding features. But as mentioned, we did not expect that the unit testing phase would need a lot of understanding to do. As a result, we may need to increase the way we organise our time to be able to complete work.

Final Thoughts

Overall, the team members were satisfied with the outcome of this sprint. The development team had a great improvement in this sprint since we implemented two features with highest priorities. Therefore, it gives us an opportunity to be familiar with the code structure, hence moving to the next sprint will require less time and effort compared to this sprint. Indeed, understanding the code requires more effort than actually implementing it.