**Wednesday May 17 Sprint 2 Retrospective**

For sprint 2 we were able to achieve a minimum than we intended to do, this is due to the backlog from sprint 1, which had a lot of work. We have less backlog for sprint 3 as compared to what we had for sprint 2, this is a sign of improvement in our communication skills and working together. The student can login as a student, which redirects them to the student dashboard, they can schedule an appointment by specifying all the required information and see all the meetings they have scheduled on their dashboard. The lecturer can login as a lecture and specify availability time slots and see those time slots on their dashboard. The aim for sprint 3 is to make sure that the lecturer can see appointments scheduled with them and the student can see the availability time slots specified by the lecturer. In this sprint, sprint points were represented by estimated time spent for user stories. Sprints velocity archived = 6.

**Sandile Kunene:**

**What went well?**

- The user can sign in and are redirected to their respective dashboard.
- The student can set appointment and see the appointment on their dashboard.
- The lecturer can set availability timeslots and see them on their dashboard.
- There is a small backlog for the next sprint as compared to the sprint 2 backlog because of sprint 1 work.
- Deployment port issue has been solved.

**What went wrong?**

- Appointments could not be displayed on the lecture side due to not linking students and lectures so that when they put the name of the lecture, the database knows which lecture it is.
- Most of sprint 2 features have not been implemented because of the sprint 1 backlog.
- Code communication from different group members is bad such that some code from one group member can block the functionality of the code done by another member.
- Other members have way less commits as compared to others, which leaves a lot of gap and imbalance work.

**What could be improved?**

- Group member collaboration and helping each other with the work we are struggling with as compared to letting the member suffer alone.
- Sign 2 group members per user story in order to enable them to help each other and keep each other in check in case one person is not doing their assigned work.
- Avoid big backlogs as this delays the work assigned for a certain sprint and makes us focus on backlogs. This will cause us to have backlogs until the last sprint, which is not good and will make our web app not ready for production.

**Karabo:**

**What went well?**

- Improved pull request reviews
- Improved and meaningful commits
- Students can schedule an appointment
- Included unit tests for user stories
- Deployment

**What went wrong?**

- Not realising dependencies earlier, therefore leading to other team members being blocked
- Starting the sprint late
- Not completing all of sprint 2 user stories. This may pose a problem in sprint 3.
- Underestimating user story days.

**What could be improved?**

- Collaboration and communication amongst team members. This was improved from sprint 1, but there is still a long way to go.
- Distribution of user stories within team members
- Completing all user stories per sprint

**Thamsanqa:**

**What went well?**

- User stories were completed on time.
- Able to deploy the website successfully
- No merge conflict as branches were short lived

**What went wrong?**

- Lack of backend development made implementing certain functions difficult
- Lack of contribution to most of the backend functionalities
- Although commits were made regularly, number of commits was subpar
- Testing has been neglected

**What could be improved?**

- Investing more time into learning Mongoose for backend development.
- Writing tests as the functions ect are created.

**Ashard:**

**What went well?**

- Lecturers can view list of upcoming appointments

**What went wrong?**

- Routing/linking difficulties between pages
- Collaboration problems due to other commitments
- All features not implemented

**What could be improved?**

- Meet with members more regularly
- Understanding of architectural layout of the website and integration
- Use automated testing of the website