Sprint Retrospective, Iteration #4

| Issue | Tasks | Task Assigned To | Estimated Effort per Task (in hours) | Actual Effort per Task (in hours) | Done (yes/no) | Notes | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Notification for matching | Rejection notification | Glenn Weeland | 10 | 12 | yes | Creating a whole notification system for this took a really long time. The communication with the other microservices was not an easy feat to just do for the first time. The notification services are bound to the user service where they can request their inbox and send replies to join messages. This issue also added the last part for the issue in sprint 3 for canceling events where the admin should be notified if a participant leaves. | |
| Notification for being accepted |
| notifications should hold details for their purpose |
| Admin accepts/refuses pending rowers | admin accepting /  rejecting requests | Glenn Weeland | 4 | 4 | yes | With the notification system in place, this became a lot easier to do. Communicating with the event microservice was a hassle for 2 hours but in the end it works correctly | |
| Sending confirmation messages on accept |
| Testing | Testing messages | Glenn Weeland | 10 | 10 | yes | Testing this much, just takes a long time. Not much else to say about it | |
| Testing Certificate |
| Testing Authorization |
| Authentication microservice testing | Create JUnit and Integration testing for Authentication microservice | Jan Bryczkowski | 8 | 12 | yes | Completely tested the authentication microservice | |
| Implemented Builder Architectural Pattern | Implemented Builder Architectural Pattern in Event, Authentication and User microservices | Jan Bryczkowski | 10 | 10 | yes | The Builder architectural pattern was implemented in the Event, Authentication, and User microservices to facilitate the creation and saving of complex objects to the database. The pattern consists of four main classes: the Builder interface, the EventBuilder or AppUserBuilder class (which implements the Builder interface), the Director class, and the actual Event or AppUser class. | |
| Finish up assignment 1  part -2 | Write assignment | Everyone | 6 | 6 |  | — | |
| Testing | Create JUnit and Integration testing for User microserice | Wiktor Grzybko | 4 | 4 | yes | A lot of stuff to test, but most of the testing is schematic so it didn't cause a lot of problems. What wasn't tested by me, was tested by Glenn | |
| Match user with events which match his availability | Create time fields in AppUser so user can set his availability | Wiktor Grzybko  Ibrahim Omar Abdalla Mohamed | 4 | 7 | yes | Initially, we tried to create a container for dates so that the user could set multiple time periods for their availability. However, this caused a lot of problems when sending the object via JSON, so we decided to use only two dates, the start of availability and the end. The rest of the implemetation was no longer problematic because the other requirements had already been implemented early on, so we only needed to adapt the filtering to one more | |
| Allow user to set availability |
| Match user with events which match his availability |
| Implement Strategy design pattern | Create implementation of Strategy design pattern, which allows the user to see events in a specific order. | Wiktor Grzybko | 1.5 | 1.5 | yes | Thanks to the tips given in the lectures, getting the pattern into our code was not a problem. | |
| Match user with events based on the event’ rules and user’ preferences | Match based on the gender restriction | Wiktor Grzybko | 6 | 10 | yes | Despite initial problems with communication and connecting users to events in the Certificate microservice. We managed to create a working and efficient matching process. Manual testing, however, required considerable time | |
| Match based on the experience restriction |
| Match based on the position restriction |
| Match based on the certificate restriction |
| Testing | Testing event | Radu-Stefan Ezaru | 7 | `10 | yes | Testing has a really slow start-off but quickly came up with it. It was the most complex class to test and unfortunately it is about 50-60% done. A lot of things were added to dev after the tests were made, and we did not have time to remake the tests. | |
| Cristian Soare |
| Certificate Hashing | Created Certificate Microservice | Radu-Stefan Ezaru | 7 | 9 | yes | It was pretty well done, however imperfect in terms of hashing index, and matching the two indexes together. Struggled a bit with adding two tables to the same H2 database, but were successful in the end. Later, some implementation choices were fixed by Wiktor. | |
|
| Create the hashing function for User |
|
| Manage the communication between User And Certificate |
| Cristian Soare |
| Manage the communication between Event and Certificate |
|
| Creating the hashing function for Event |
|

Project: Rowing

Group: 34a

Main Problem encountered:

* **Description:** Multiple merge conflicts appeared while coding and testing at the same time, since functionality was added/changed during the last sprint
* **Reaction:** We assigned one person to solve the conflicts and manage to push correctly while the other would fix the issues coming from merging