Sprint Retrospective

Iteration 1

Project	Stereotypes in CS
Group	10D

Done	Partially done	Not done	Removed

User Story	ID	Status
As a project owner, I want a specific quiz format	1	
As a project owner, I want the results of every test to be stored in the database so that they can be retrieved afterwards	2	
As a project owner, I want the data to be securely stored in the database according to GDPR regulations	3	
As a participant in the study, I want my guardian to be able to complete a consent form so that I can take the test	4	
Tasks that are not necesarrily related to a user story, but which are important for the whole devolpment process	5	

User Story	Task	Assigned to	Estimated effort (hours per person)	Actual effort (hours per person)	Done	Notes
5	Set up project structure	Everyone	5	7	Yes	Initialize basic project structure; install all dependencies and tools.
5	Set up CI for front-end	Dragos	1	2	Yes	Use Gitlab CI feature in order to check deployments. Currently CI has three stages: lint, build and test. More rules are going to be added to lint stage
4	Create consent form	Andrei & Dragos	12	15	Yes	Fully functional consent form; send and receive information from the database.
4	Allow multiple children in the consent form	Andrei & Dragos	3	3	Yes	A parent can register multiple children for the experiment.

1	Create data collection application	Andrei & Dragos	20	20	Yes	Basic functionality for the sprint set up. The Quiz can only be started with admin rights. The loaded quiz is still hard-coded, and is set to be properly synchronized with the database during the next sprint. The workflow for a quiz is clear and the application overall is stable. Polishing the application is scheduled for early next print
1	Style quiz	Andrei & Dragos	6	8	Yes	Quiz app is mainly styled using third party module Material-UI. Where necessary, css was used. Overall, the quiz is styled and in compliance with the UX requirements.
5	Refactor code	Andrei & Dragos	4	5	Yes	Code refactoring and documentation. Refactoring has been done repeteadly after each feature was implemented. Most of the refactoring involved wrapping duplicate code in React Components for modularity and good use of design patterns (store, hoc)
5	Test front-end using Selenium	Andrei	2	3	Yes	The application is tested using Selenium.
5	Test front-end using Enzyme	Andrei	3	3	No	Basic tests are currently implemented, but significant effort is required to ensure proper validation. However, research has been done and the implementation of the tests is going to happen early next sprint
2,3	Setup a connection with a remote database	Alex & Alin & Ionut	2	3	Yes	Add a Postgress database on Heroku and connect it with our Flask API.
5	Setup CI/CD pipeline for backend	Alex & Alin & Ionut	3	3,5	Yes	Add several jobs to our pipeline, including running tests and static analysis tools. We also have jobs that we can manually trigger to deploy our application to Heroku.
3	Add an authentication scheme for an endpoint	Alex & Alin & Ionut	4	6	Yes	We have added an authentication scheme using JWT tokens. Tokens can be obtained in order to access certain resources from our API.
2, 3	Setup the database	Alex & Alin & Ionut	2	1,5	Yes	Add the Flask SQLAlchemy extension to our Flask application. Create and add models to store Users in the database. Add some basic rows to the table.
2, 3	Setup database schema for IAT	Alex & Alin & Ionut	4	5	Partially	We have created different database schemas to fulfill the requirements of our application. Before committing to a certain one we will first discuss them with our client
2, 3	Add migration tools for the database	Alin	1	1	Yes	Added the flask-migrate extension to our application so that we can more easily change schemas. Now every time the schema changes we run two commands to update the database.
2, 3, 4	Insert consent form object into db	Alex & Alin	1	1,5	Yes	Take the consent form information from the frontend app and store it in the database. The signature is momentarily stored as a base 64 image, although we are looking to move to a image hosting website such as Cloudinary during the next Sprint
4	Serve consent page from the server	Alex & Alin & Ionut	1	3	Yes	Created two possible ways of serving the consent page from the server. The first variant uses a different server that serves static files. The other uses the Flask backend to send the files.

5	Add Pytest	Alex & Alin	1	2	Yes	Add the Pytest plugin to our Flask backend and add unit and integration tests for our API.
5	Fix static analysis errors	Alex & Alin & Ionut	1	2	Yes	Once we added the Pylint static analysis tool we fixed the numerous error it was giving to us. Most of it was about writing documentation.
5	Refactor API code	Alin	2	3	Yes	Refactor the backend code to make use of the flask-restful extension. Also added a separate blueprint to increase code modularity. After this was done it will be easier for multiple people to work at the same time on the API
5	Use factory pattern to create Flask app	Alex & Alin & Ionut	1	2	No	We tried to use the factory pattern as described in the Flask docs because we believed it would increase our code quality. We ended up not doing it because in the end we felt it made things more complicated
5	Editing and uploading meeting notes	Ionut	1	2,5	Yes	

*Technical writing assignments, meetings and other organizational tasks are not included in the retrospective

Main Problems Encountered

Problem 1 ***Uncertainty in requirements***
Description Content of consent form or specific questions from the Implicit Association Test was (and for the most part still is) missing due to the delay in the researcher's activity
Reaction This had obviously a negative effect on the development of our application, since it involves performing a task and changing it at a later point in time in order to fully respect the requirements.

Problem 2 ***Estimating time per task***
Description On average, our time estimates are below what we've actually spent on tasks.
Reactions We looked at what was the cause for this issue and realised we don't take into account the time it takes to document ourselves on a certain piece of technology before starting to use it.

Problem 3 ***Heroku deployments***
Description Unfamiliarity with Heroku platform caused some delays
Reactions Some trial and error debugging which took more than expected was necessary to discover the causes for unsucessfull deployments.

Problem 4 ***Difficulty in choosing between two different implementation appoaches on some tasks***
Description At the beginning of the project decisions have to be made regarding how we will implement certain things (such as the Login for example)
Reactions In some cases we spent too much time discussing between the different approaches we could take before taking one, but since we only have to do them once in the beginning, it will only affect this sprint

Problem 5 ***Displaying code coverage percentage and linting score in README.md***
Description Due to not being project admins on GitLab, we were not able to successfully add code coverage and linting to our README.md
Reactions We fixed the linting score with a workaround, but the code coverage percentage is still broken.

Adjustments for next Sprint Plan

1. Communicate more with the client and try to understand all the requirements in advance.
2. Research more how IAT tests work.
3. Do take into consideration the learning time estimate when working with something new.
4. Try to split larger tasks into smaller parts that can be divided evenly between team members.
5. When creating the sprint backlog take into consideration the amount of tasks that still need to be done by the end of the project.
6. Take care of static analysis errors while coding something and not after all the code has been written.
7. Start testing early. Try to write test right after implementing a new feature.
8. Put more emphasis on documentation.