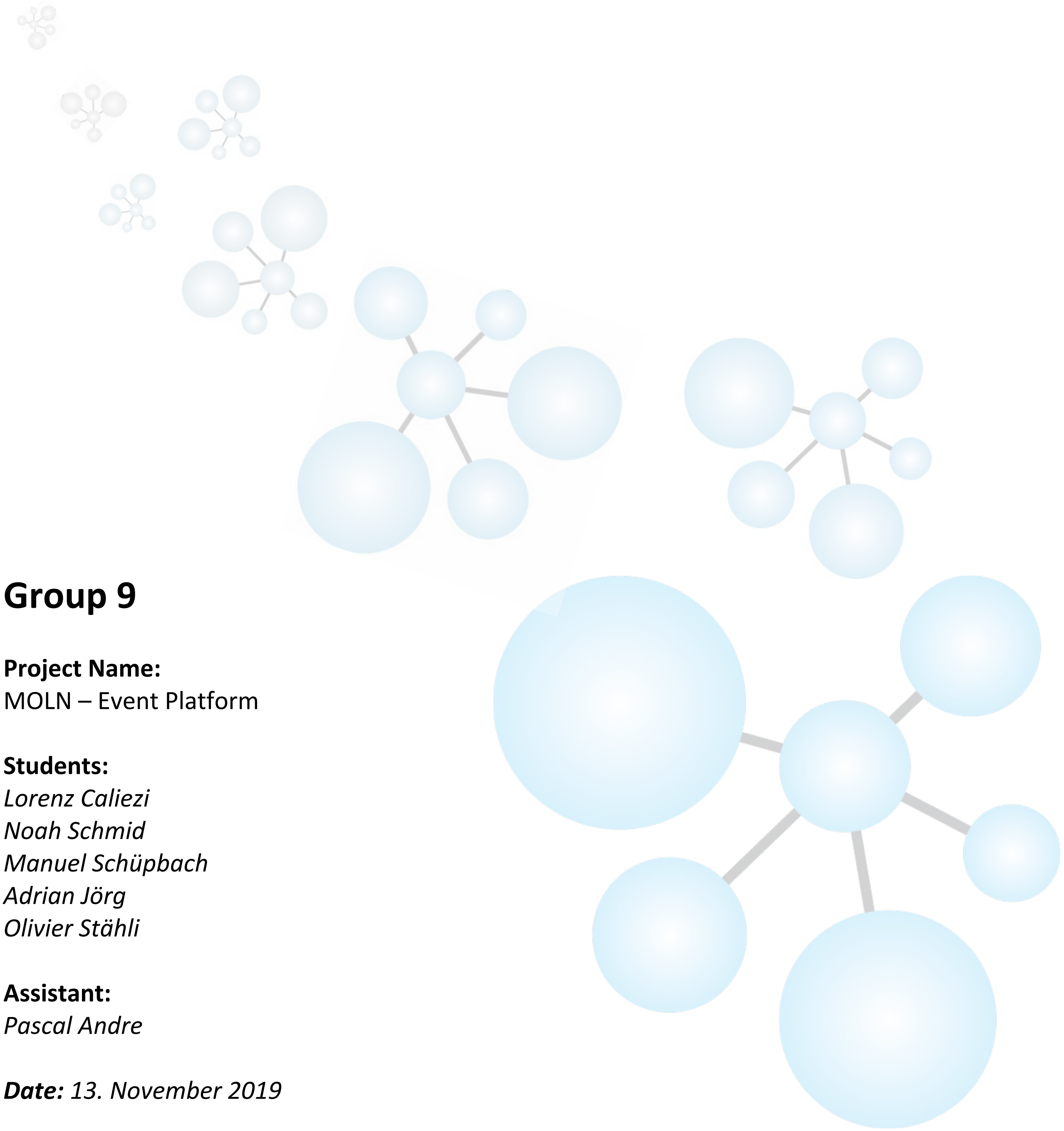
Biweekly Report III

Introduction Software Engineering



Group 9

Project Name:
MOLN – Event Platform

Students:
Lorenz Caliezi
Noah Schmid
Manuel Schüpbach
Adrian Jörg
Olivier Stähli

Assistant:
Pascal Andre

Date: *13. November 2019*

Table of Contents

General

Adrian joined our team on 30. October 2019. He is a very resourceful addition to our existing team. He helps Lorenz and Olivier with the frontend and has already brought many new inputs regarding project structure, best practices and code style.

Major Challenges

Solved

- Integrating a new team member:
    - Distributing work and responsibilities
    - Familiarize Adrian with our project and workflow

Frontend

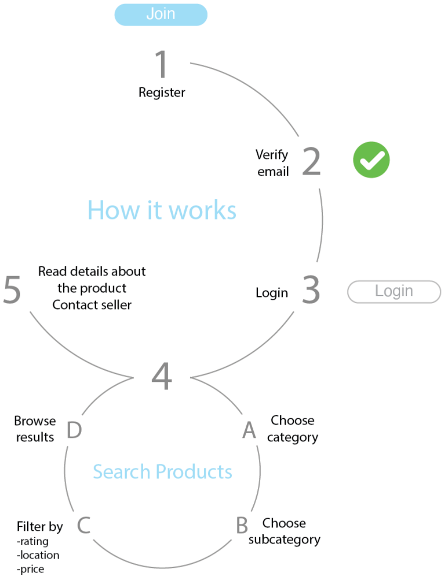


Figure 1: This shows the user the process from registration to contacting the seller to buy a product

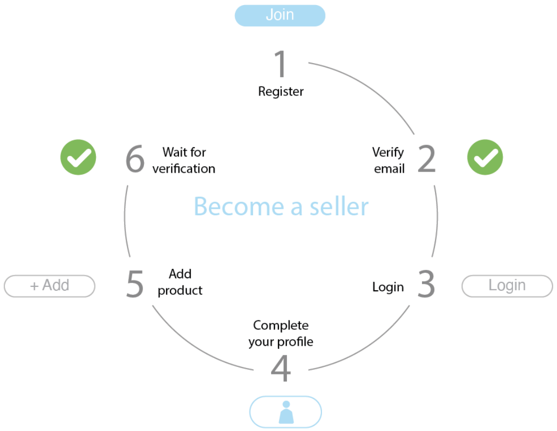


Figure 2: This figure illustrates how one can become a seller with a verified product on MOLN.

Design

Implementation (Status on 11. November)

Week 1:

- Implement admin panel (delete offer)
- Refine category carousel on homepage
- Add routing between homepage, subcategories and products
- Re-implement code for the navigation-bar
- Restructure and rename basic design of the project:
    - top-header → header
    - middle-header → home-banner
    - bottom-header → navigation-bar
- Styling

Week 2:

- Implement admin panel (manage user)
- Add profile page for users
- Filter function in subcategory page
- Dropdown menus for subcategories
- Add Compodoc documentation
- Reach a 25% documentation coverage on our code (frontend)
- Add functionality such that user can add products
- Styling
- Add infographics to homepage

Major Challenges

Solved

- Finding a good framework for the documentation

Pending

- Writing the documentation/ commenting project
- Code refactoring
- Adding a lot of data such that our website makes sense (diligent work)
- Making our application look nice on iOS

Learning Outcome

We learned:

- That Compodoc is a very useful tool to create a proper and visualized documentation (shout-out to the Angular pros from Inpeek: Simon Christen and Fabian Küng)
- How HTTP requests actually work

Backend

Implementation (Status on 11. November)

Week 1:

- Changed the category system from a nested system to one based on materialized paths. Thanks to this adjustment, we don't have to save categories multiple times. Editing and deleting categories get much simpler and faster.
- We did a lot of cleaning up as we learned new ways of implementation.
- Rewrote all tests using chai as request and assertion framework. Implemented the before/after Hooks of the tests.
- Made the buildup and teardown function used by said hooks

Week 2:

- Added support for uploading and storing images on the backend.
- Optimized performance of many controllers.
- Implemented authorization features, so that only admins can edit every value. Users can now just edit their own profile and their products.
- Images are getting deleted as soon as they're not needed anymore.
- Started implementing the notification and order system, which will be needed when implementing placing orders. The notification system will be needed to notify sellers that their product was bought, or whether it was accepted.
- Comments/ratings can be added to products now.
- Wrote tests covering user, product, and category actions. The user is fully tested, but product and category have some untested requests.
- Shown below is the build, testing and cleanup of a testcase.
- In the real test there are more tests being made on the returned user. But they are omitted in this example. Note that the blue Arrows represent a request of which the response is being tested.

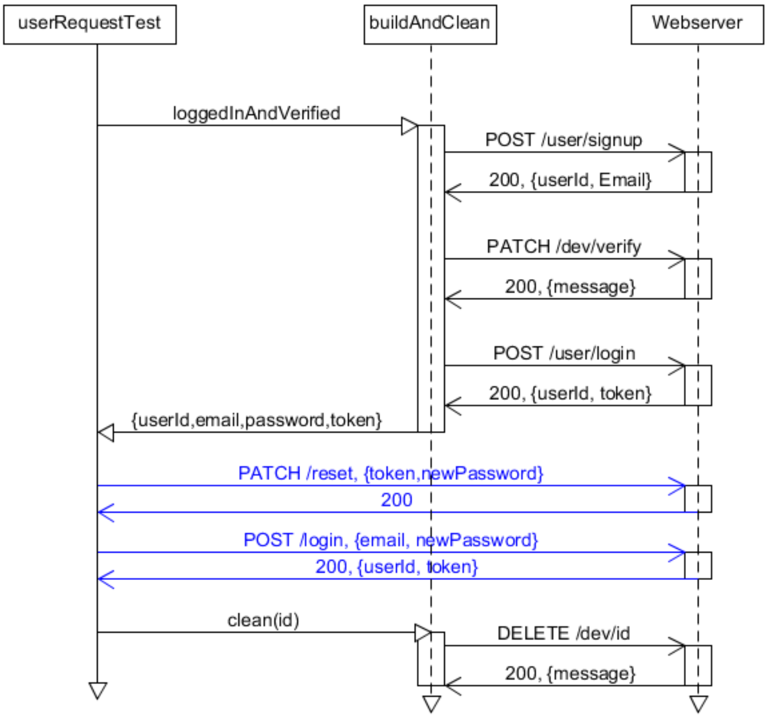


Figure 3: This example shows how the reset password request is being tested. Note that all requests utilized by buildAndClean are being tested before being used here.

Major Challenges

Solved

- Coordinating changes with the frontend developers so we don't mess everything up. We had to change some things in the backend which resulted in a different display of the information. It makes more sense the way it is now but it resulted in the frontend team having to change the handling of the backend calls.
- It was hard testing the requests because of the asynchronous nature of node js. This made debugging pretty hard because the mistake could be in the test or the actual code. However, using promises and async tests the problems were solved.
- Since the code was being changed during test development, some time was lost by modifying the tests for the new responses.

Learning Outcome

- We learned many small things about MongoDB which improved the code a lot.
  - For example referencing other documents and the populate() command, which was done by hand in the past (this caused a drag on performance).
- Learned how to setup tests and how assertions in javascript work.