Within this module we had to deliver several objectives. The main objective of this module was to design and code a secure app that encrypts files that users can upload, share, delete and update in a web environment. This task took a lot of time, I am however nonetheless great full that I invested that much time into it, since I got thorough understanding of the matter by experiencing the challenges firsthand.

Initially we planned to have various features within the app to exceed the goal we were given. Since we were planning on using an agile process, we luckily defined all must have criteria as part of the minimal viable product (MVP). All secondary features to make an even more secure and nice-looking app were not listed within the MVP. Having an MVP defined and working on this first came in handy as we realized that we underestimated the task of coding a lot. We soon realized that we might only make the MVP in time and that we must skip all additional features.

But not only did we underestimate the task we also picked a non-feasible approach at first, since we initially tried to use an already published app where we planned to “just” add file sharing and encryption functionalities. We soon had to realize that this task was not easy at all, since our lack of flask skills made it very hard to read and understand the already written code. Eventually we figured out that this approach will not provide the MVP. After having realized this, I started to code an app from the beginning by following a tutorial and adding very basic stuff, like forms, MySQL backend connection etc. step by step. By doing it this way I had to troubleshoot a lot of code in total, but at the beginning the code was rather small and I was able to circle the issues and fix them one by one. By doing so I more and more got to understand the syntax of flask and jinja2 and the codebase grew by the day.

The plan to code an app from the beginning was a great decision. Not reinventing the wheel on the other hand however was still very important, since we had to deliver within time. Luckily Flask has many very powerful libraries to keep your code small and understandable, but still rich of features. A good example to mention here is flask-login, which provides you with cookies for the client and an object server side, that takes care of authentication and provides very easy authorization checks within your app. This module took quite some time to integrate and to understand. Coding the functionality, myself however, would have taken way too much time.

Having coded HTML, PHP & JavaScript apps before that needed some not common backend functionality, I personally very much enjoyed my moment of breakthrough when Jinja and Flask were understood at a level, so that I was able to easily code Python even within html documents itself. I literally was amazed to see that any Python code, not intended to be part of a web app at all, such as Fernet, which is part of a rich python cryptography library, for example could be integrated with ease into a web app. Due to this easiness of integration we realized with our app, that files are not only encrypted, but we realized that the files are encrypted with a very strong cipher suite (AES128 CBC).

The choice to pick flask was great. The choice to plan for loads of functionality in the beginning wasn’t good. We luckily however adapted quite well and delivered not just an MVP, but a very secure MVP after all. Testing we conducted also did not revealed major security flaws. Out testing however omitted a penetration test, which would be necessary to conclude a full development circle. Due to time, this could however simply not be delivered.

The app however still maintains its flaws since some functionality was nonetheless coded in a rush. I for example coded two get request, that don’t check authorization and don’t provide or verify any CSRF tokens. Those flaws might only be visible on the 2nd look, but they might lead to severe issues, if this system would be used in the real world.

I know do truly understand one of my colleagues at work that has 10+ years professional coding experience, when he says that in some companies the lack of security doesn’t come from lack of understanding but from lack of time, since I have made exactly this experience myself now. With sufficient time given to the developers most software would be more secure than it is. Since resources however aren’t infinite, a methodology such as security risk management is key to address the issues having the most risk first, to code a most secure version of an app within a given time.