CSE 545: Secure Banking System Individual Report

Vishnunarayanan Ramasubramanian*
ASU ID : 1213169553

1 Introduction

The aim of this project was to implement a skeletal banking system with industry-grade security. A requirement specification document was provided with a detailed description of the application, along with functional, non-functional and security requirements.
As security and quality assurance was one of the main focuses of the project, our team focused on ensuring that all or at least, most of the security requirements were taken care of from a very nascent stage in the software development life cycle. However, in the design phase, our team faced a lot of issues. Some of the major issues we faced were in deciding what web application framework to use. The main candidates were unsurprisingly, Django and Spring MVC. Both frameworks were well renowned in web development and offered various features with regards to security. Since most of the team felt that they were more comfortable with coding in Java over Python, we decided to go with Spring.
In retrospect, we probably would have benefitted more with a Django application over a Spring based one, as we later discovered after the design phase that Django provides more specific native security functionality to ward off against attacks like XSS, which were one of the attacks that later turned against us.

2 My Contribution

My primary contribution in this project spanned from the design phase till the testing phase. I was assigned the task of implementing parts of the core functionality. In particular, I was reponsible for developing functions that interacted with both the route layer and the database layer, like transaction and technical account access, credit and debit card functions, and banking statements. Since I was responsible for a layer that interacted so closely with the database layer, I aided Anjani Sai Chatla and Apoorva Venkatesh with the design of the database schema wherever I could.
Though the functionalities that were assigned to me were fairly straightforward in understanding, it still took a while to get them up and running because of a few snags our team faced along the way. This was because setting up the Database and the MVC layers took time as the entire team was quite new to the Spring framework as such, and we were still facing a few more design issues in some layers of the Spring framework.
One of the most significant issues that we as a group faced was the dilemma between using Spring and SpringBoot. Though Spring offered most of the relevant features that our application sought, SpringBoot offered much more in terms of code readability and ease of coding. Since the tradeoff between these two frameworks were too close, it took time to arrive at a consensus. Due to these issues, our team started to face difficulties with respect to the development plan in the given time frame.
Once these functionalities were done, I did the front-end design of the entire application. The front-end was written in HTML in the respective JSPs by the other members of the group, and I designed User interfaces on top of this layer so as to develop a presentable, user-friendly application. I accomplished this using the CSS and JavaScript framework, Materialize. This framework was based on Google's material design template. I also used Google's material icons in the interface to make it easy on the eye. This was a major contribution to my learning curve involved in this project, for I had no prior experience in front-end design and I thus had to learn this from scratch in this project with some help from Naveen Piedy. I also helped Naveen Piedy in the integration of the Virtual Keyboard in the front-end.

*e-mail: vramasu2@asu.edu

3 Results

In the end, due to the efforts and motivation that our team lead, Anjani Sai Chatla and deputy leader, Zarah Khan gave, we were able to successfully deliver a robust, secure banking application within the stipulated deadline. Though our system did have a few potential pitfalls against some basic attacks like XSS due to a little oversight, the application as a whole worked as expected for the most part and the number of successful attacks on our system were relatively few. Our team was dedicated to this project in the sense that we held periodic, productive meetings every Saturday, where we discuss the progress of the group as a whole and address any difficulties a member of the group faces while implementing the work that they were assigned. Since there was continuous communication between all the members of the group with respect to the progress in the project, the team knew where we stood with respect to the Gantt timeline and it became easier to plan and decide what had to be done next.
Despite having such a high level of communication, like with any project with a large team, errors and oversights tend to happen. This is the reason why our application was found to miss some necessary functionalities, such as viewing balance. Though such mistakes tend to happen, we as a team should have double-checked all our requirements and made sure we exhaustively included all missing cases in the user manual. Errors like these tended to happen because of the time crunch we were facing, some miscommunications and oversight, all of which are quite unacceptable excuses in the industry.
Some basic attacks managed to succeed in our system, such as XSS, which was because of improper input validation. Since the work was split across teams in different layers, the likelihood of such lapses increased which when combined with lack of proper communication turned out to be a perfect recipe for disaster in the final application. Broken functionalities needed to be fixed and missing ones had to be incorporated into the user manual at least, if not integrated into the application.
Despite this, most of our core functionalities turned out to work reasonably well. For instance, when attempting a transaction, if a user gives an invalid amount or AccountID, the transaction does not go through and a message saying the transaction was unsuccessful was displayed. Such measures ensured that the user knew what exactly was going on and aided the user in identifying how to correct it.

4 Conclusion

Though the team tried its best to follow the Work Breakdown structure and the design documents to the letter, due to some of the aforementioned issues we faced in the design level and the time crunch, we had to improvise at places. Thus, though the breakdown was thorough, the abstraction of work allotted to different teams in the group ceased to exist and we all ended up helping each other

with whatever each of us could bring to the table to ensure that the application was delivered on time with the requirements met.
Due to various constraints that came along with developing such an industry-grade project with a large team, we all had our differences as we had people from various degrees of experience in designing and developing such a security-based system. Despite that, we all came together to work as a coherent unit to develop this application, which, in my opinion, turned out to be a great learning experience. As a person who had never been exposed to any sort of security-specific domain knowledge, the learning curve was particularly steep for me from both this project and the course. I got to learn about the various practices and standards that are employed in the real-world, along with the most common attacks that any application faces. I hope to thus get a chance to apply all of these practices in my future endeavors and I believe that this course has gotten me a concise and necessary introduction to the study of security of software and its practices.