1. Contents

[1: Introduction 2](#_Toc528184012)

[2: Dependability 3](#_Toc528184013)

[3: Security Engineering 4](#_Toc528184014)

[4: Resilience Engineering: 5](#_Toc528184015)

[5: Conclusion 6](#_Toc528184016)

# 1: Introduction

The developing of software is not just developing and adding functionality. A big part of the work in software development is not visible when it is released and this is also how it should be for some parts. Other parts of the software like the functionality itself and user interfaces should be visible and to keep them visible it needs support and defense. Dependability and Security are, especially in very delicate systems, big issues. These systems have to run without an error, without any security issues and this for a long time without a restart in between. Due nothing is a hundred percent safe in software engineering an error may occur nayway but then, when the worst case becomes true, the system has to be able to be recovered quickly without losing important data or getting into inconsistent states.

In this paper we will analyse what properties in terms of security and resilience engineering our product should have, if it has them and how they are implemented.

# 2: Dependability

During the implementation of our first version of the game we just developed functionality and interface. We did not spend time and effort in thinking about making things safe or stabile to work. Basically we also do not have to. Since we are developing a mobile app for android we can use all the offered functionalities provided by the play store. That is on the one hand much cheaper than developing it on our self and also much more reliable since our knowledge about programming secure and dependable software is not very big. The extra time to learn it properly will cost to much money and we will need longer to develop updates afterwards. Especially for a mobile game this is unnecessary and overdone.

# 3: Security Engineering

Our System also have nothing like security standards against hacker attacks. If somebody want to crash our app it wont be to difficult for him or her. But why do we not care about it. First of all, our mobile app is based on the device of the user who uses it. If this user decides to crash the game on purpose only he himself can not play it anymore. Other users are not infected by that. If a user really want try destroy the app for all users he would have to be able to write a fake update, pretend to be us and upload it to the play store and wait for others to download it. Since there is no transaction system, log in system or anything else dealing with sensible data of users or money at all, this hacker wont get anything from that and would waste his time. Also google security standards should be very high and nearly impossible to hack if you are alone and there wont be a whole organisation trying to bring down our mobile game.  
 So what instead of using our app as an entry to the whole system? This could be a problem, but as said before if a user wants to steel data from another users phone. He would have to manipulate the current version in the play store to make other users get an unsafe version. In this case a smartphone will always ask the user if an application get the permission to use special thing on your phone like the contact list, the camera and so on. If a hacker is even good enough to break this security wall, than he also would be good enough for a bigger app with more potential in making money or he uses an app where I have to give permission for accessing contacts and data on the phone and SD-cart anyways. Our app does not require contact data or voice recording or camera use from the user and it would be odd if it did. So to sum it up, our app is not really a target for an attack, wont bring much possible profit for the attacker/ damage for the user and therefore does not need much effort in security engineering.

# 4: Resilience Engineering:

We covered in previous chapters why we wont spend effort and time in security and dependability issues, because the risk for something bad happening is just too low to be worth it. Since there is always an exception from the rule there still can be an attack on our system or any other issue that decrease the performance for the user. What can happen and how do we plan to recover from it. First scenario is tat the app runs really slow because of not enough RAM in the phone so that it has to outsource important data for calculations to the hard drive/ SSD/ SD of the phone. This could happen if the user has to many other apps in the background running or not enough space on the phone at all. We cant change the situation and of course we can not just delete random data or kick other processes from CPU and RAM. The user simply as to deal with that or delete something on his own. Our app wont be the only one effected by that and since our app does not need much resources it should not be the first which get delete if the user has to make a choice. Second scenario is no connection to the internet. If our app already reached the multiplayer version we spoke about in previous papers this would reduce functionality. There is still the single player mode and depending on our progress our offline AI so the user still can use the app, an advantage over other games which can not be played without internet. The next worse case is that a hacker can manage to upload a corrupted version of the game to the play store and the user downloads it. In this case the user could use the app to steal other data, delete it or maybe even use the phone as a bot. Those are the worst case scenarios. One possibility to avoid this could be to check always if the app is on the current version provided by the owner when opened. A smart hacker would change that to not check that so it might be not really helpful. So the user has to deal with that and basically we can not do much to protect the user from that scenario, but since all these security standards of the play store are really high we do not have to be afraid of this. What we can do is to try to recognize a wrong version as fast as possible and update it back to a working official version and send a message to all the users via the play store to update it again. Beside the fact of possible loss of data we could lose users by that. Some users who would delete the app immediately if it is not working, some others would report this over the play store rating function or the support function in our app if this is already done and still working. The user data (i.e. progress in the champagne) can either be saved on the users device or in a cloud provided by the play store so the data would be safe unless the hacker manipulated it so that the app deletes the progress. In this case it wont matter where the data is and the user would have to start from the beginning after our app is recovered. There are not really any other threats like a black out, denial of service attacks. Even bad user input is not really an issue since only the elements we provide can be placed in the game and there are no entries like in text boxes. To make it short, there are no “normal sized” issues. Only the minor ones which do not effect much and are not important and the really big ones from which we just can not be a hundred percent safe just like any other app can not.

# 5: Conclusion

In conclusion we showed that Dependability and Security is not really a big topic for us. Maybe it could become on in case we will integrate something like micro transactions and a payment system in our app but until then our app is free from sensible data. All security functionalities are provided by the platform we want to offer our app on and also payment transactions and saving user data can be outsourced to it. Also the fact that our app is quite small at the moment and wont have that many users, i.e. facebook and whatsapp, from a realistically point of view, safes us from the most trouble.