# Introduction

## Background

Throughout the last ten years, mass cyberattacks against major organizations have amplified with direct impact on the organization’s finances and data leakage. Security breaches are the most prominent cause for attacks being allowed to happen. Although different types of organizations become victim of cyberattacks, an analysis on data breaches experienced in multiple organizations established that medical organizations and BSOs are the least prepared against attacks . Furthermore, the latest reports confirm that almost half of businesses have been confronted with cyberattacks since companies are settling into the new normal. Cyberattacks have grown in frequency and severity since the pandemic . Lallie, Harjinder Singh et al observed that there appears to be a loose correlation between the announcement and a corresponding cyber-attack campaign which utilizes the event as a hook to increase the likelihood of success . Thus, to minimize or eliminate the damage caused by cyberattacks, software security has to be addressed during software development.

Security breaches are caused by security vulnerabilities in source code introduced by software developers when creating software, and therefore developers are often blamed for vulnerabilities . However, application security is primarily performed by security experts causing a separation between security and development. As a result, the probability of insecure software is increased .

Writing secure code is critical with the prevalence of security vulnerabilities. To achieve this, developers need to be aware about the potential vulnerabilities they might introduce when developing software features and understand how to mitigate them. Still, the knowledge and skills to produce secure software is lacking and often are not taught in computing curricula despite the existence of secure coding guidelines .

Such secure coding guidelines are provided by the Open Web Application Security Project (OWASP). For instance, the OWASP Top Ten brings awareness about the most common security vulnerabilities found in software. Furthermore, security standards such as the OWASP Application Security Verification Standard (ASVS) and the Mobile Application Security Verification Standard (MASVS) enables organizations and developers to produce and maintain secure software.

To address security during software development, this study will implement a secure agile Software Development Life Cycle (SDLC) with the OWASP Security Knowledge Framework (SKF) to develop CheFeed, a mobile application developed for this thesis. SKF is an expert system application that uses secure coding guidelines such as the OWASP ASVS and OWASP MASVS to assist developers in pre-development and post-development phases to create secure by design software.

## Aims and Benefits

### Aim

Security vulnerabilities found in applications introduced by software developers are the cause of many security breaches and data theft. To reduce the introduction of security vulnerabilities security must be addressed during software development. The aim of this study is to implement a secure agile SDLC with testable security controls configured through SKF.

### Benefits

This study will provide insights into the process of developing software in an agile SDLC where security is addressed from the beginning. It provides a clear presentation on the benefits of using SKF to configure feature sprints where security should be in place. Students and software developers may benefit from understanding how secure code guidelines can be used. Furthermore, the implementation of SKF for a secure SDLC could serve as a tool for future studies to build upon the process presented in this study.

## Scope

The scope of this study is limited to configuring MASVS Level 1 (MASVS-L1) security controls with SKF for the development of CheFeed. MASVS-L1 ensures that mobile applications cohere to the best security practices concerning code quality, handling of sensitive data, and interaction with the mobile environment.

This project is a group effort where two fellow students will contribute to the development of CheFeed. Briefly their study contribute to the following:

* The development of the API and database design by Stephanus Jovan Novarian in his thesis .
* The development of a sentiment analysis model which uses recipe reviews as its input by Ikshan Maulana in his thesis .