Comparative Assessment of Web Frameworks' Security Features Through OWASP Guidelines

Applied Research Project

by

Anastasiia Tiurina (22401247)

School of Innovation, Design, and Technology

Wellington, New Zealand

2025

Abstract

asdfgf

Contents

Glossary

API

DOM

Introduction

Motivation

As time goes on, our everyday lives become increasingly intertwined with technology with each passing moment and every new service we engage with or account we sign up for. As this process continues, more of our personal and sensitive data ends up on the servers of companies. This factor raises important questions about the security of these web applications and services, as insecure data storage or improper handling of data can result in data breaches, corruption, or loss of information.
For instance, in their work (REFER: One Data Bre) analyse the Capital One data breach, which influenced more than a 100 million people in US and Canada, exposing their sensitive personal data, such as name, addresses, credit scores and limits, bank account details, Social Security Numbers etc. (REFER: in Open Source Kuber) mention Uber data breach that compromised personal information (PI) of more than 50 million people – it was caused by misconfigurations in Kubernetes.

In cases like mentioned above, a thorough understanding of the tools – referring here to web frameworks and libraries – and technologies used by developers during implementation is crucial. When addressing security in web applications it becomes clear that cybersecurity is a multifaceted subject influenced by a great number of factors – including the development practices adopted, user behaviour, choice of implementation tools etc. The development practices, however, are strongly influenced by selected tools, and if a developer does not understand the intricacies of how the tool works, it might cause problems in security domain of the project. Among other factors (REFER: why do dev) mention giving a priority to functionality first along with time restrictions, difficulties to overcome the steep learning curve and – interestingly – a general lack of initiative or motivation to implement security mechanisms, often simply because developers were not explicitly asked to do so. (REFER: less is more) confirm the latter in their work. Additionally, (REFER: less is more) complete the outlook on developers' perspective saying that incorporating security practices into software development lifecycle processes, like code reviews, is challenging because it requires

substantial security knowledge and motivation, which many developers lack the time or training to acquire. there are a lot of frameworks and technologies built around the JavaScript programming language, and it can be confusing for developers to orient themselves within this ecosystem (REFER: first 20 yea). Consequently, security vulnerabilities still frequently slip through code reviews into production code.

In recent days, a significant number of developers – and those learning to code – have been utilising AI, such as GitHub Copilot, Cursor, ChatGPT and Claude, to write and understand code. These tools are gaining popularity due to the advantages of reducing the manual effort to implement code by partly automate and speed up the code writing process, and developers can do it using natural language (REFER: ChatGPT Generated Code, REFER: Live Programming).

Research Problem and Research Question

sdfsdf

Literature Review

OWASP Top 10

Sssdf

Frameworks

Asdf

Vue.js

cvbcvb

React.js

sdfs

Methodology

Benchmark

Xzx

Scope and Chosen Frameworks

The world of web applications is highly dependent on JavaScript and, as (REFER: first 20 yea) state it is one of the widely used programming languages with a vast number of frameworks, and this number continues to grow. Consequently, the community survey (REFER: Stack) had JavaScript frameworks at the top of the list and Vue.js and Next.js frameworks became candidate subjects for current research.

The original endeavour was to explore the work of two frameworks – Next.js and Vue.js. However, the choice of the frameworks shifted to React.js and Vue.js due to the findings discovered during implementation of web applications. While React.js and Vue.js are both frontend frameworks, Next.js is a web framework build on top of React.js. (REFR: Thakkar) Its main feature is rendering of web page on the server where the web application is hosted. Next.js has a toolset that is significantly differs from React.js or Vue.js, that handle rendering on client side (REFER: E-bu). Therefore, Next.js possesses broader API for processing data which a priori means wider coverage of potential vulnerabilities of web application it can handle.

React.js and Vue.js on the other hand, both component-based and use data-binding mechanisms to abstract over UI elements through declarative code. They both rely on virtual DOM but work with it differently and as mentioned before, work on the side of a client (REFER: millio, E-bu).

Front-end frameworks play a significant role in the users perceive the resource that they use. That being said, sdfsdfsdf state that

subtitle

Some text heading three

Some more text

Bibliography