Report Title: E-com website with 2FA and Real Time Transaction CSCE 5560 Secure E-Commerce course and CSCE 5550 Introduction to Computer Security

Ali Zarafshani, CSCE, University of North Texas

|  |  |  |
| --- | --- | --- |
| Sukanya Kolli  *MS in Cyber Security  (of Affiliation)* *University of North Texas  (of Affiliation)* Denton, USA  sukanyakolli@my.unt.edu | Naveen Ajay Karasu  *MS in Cyber Security (of Affiliation)* *University of North Texas (of Affiliation)* City, Country naveenajaykarasu@my.unt.edu | Jyothika Raj Samineni  *MS in Computer Science (of Affiliation)* *University of North Texas (of Affiliation)* City, Country jyothikarajsamineni@my.unt.edu |
| Kavya Kunduri *MS in Computer Science (of Affiliation)* *University of North Texas (of Affiliation)* Denton, USA kavyakunduri@my.unt.edu | Pravallika Pandaraboina *MS in Computer Science (of Affiliation)* *University of North Texas (of Affiliation)* Denton, USA pravallikapandaraboina@my.unt.edu | Pawan Kalyan Puligowni *MS in Computer Science (of Affiliation)* *University of North Texas (of Affiliation)* Denton, USA pawankalyanpuligowni@my.unt.edu |

***Abstract***— **The proposed project aims to address the expanding hazard of unauthorized get to and extortion on websites by actualizing secure watchword arrangements, 2-factor verification, and installment door integration. The venture will implement solid and interesting passwords, include an additional layer of security to the login handle, and coordinated a secure installment portal that complies with industry benchmarks such as PCI-DSS. The venture is anticipated to improve the security of the framework, ensure delicate client data, increment client believe and certainty, and avoid negative impacts on client believe and location notoriety caused by extortion and breaches of delicate information. This report analyzes the issue, proposes an arrangement, and anticipates positive results for the framework and its clients.**

I. INTRODUCTION

In today’s world of improve in technology, online transactions have become the most popular and most commonly used feature. It makes easier and most convenient way to shop, to bank and to conduct business from anywhere. As the technology is improved, there are also many disadvantages with the increase in technology. There is a risk of unauthorized access and fraud activities. Sensitive information like credit card details, personal information, and login credentials are vulnerable that makes hackers easy to attack. This may lead to financial losses. Moreover customers are becoming more aware of this type of risks and are expecting businesses to take steps to protect their data and ensure a safe online experience.

To address these concerns, we propose implementing secure policies like 2-Factor Authentication and Payment Gateway integration. Implementing these security measures will improve the security of the system, protect sensitive information like user information, which improves user trust and confidence that prevents negative impacts on customer trust and site reputation caused by frauds and breach of sensitive data.

II. MATERIALS AND METHOD

1. *Technology USED:*
2. *PYTHON:* Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.
3. *Flask Framework*: Flask is a micro web based framework written in Python. It is classified as a micro framework because it doesn’t require any particular tools or libraries to run. It has no DB abstraction layer, Form checking, or any other components where third-party libraries provide common functions.

Flask supports the use of extensions that can add various features as if they were implemented by Flask itself. Extensions exist for SQL Injection, form verification, file upload handling, various other open authentication technologies and several common framework related tools.

Applications that use the Flask framework include Pinterest and LinkedIn.

1. *HTML: Hypertext Mark-*up Language (HTML) is the standard mark-up language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web.

Web browsers receive HTML documents from a webserver or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages.

With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

1. *CSS*: CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate ‘.CSS’ file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it feasible to present the same mark-up page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device. The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

5) *SQLite3*: SQLite is an in-process library that implements a self-contained, server less, zero-configuration required, transactional SQL database engine. The code for SQLite is open source and available in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects.

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database file format is cross-platform means you can freely copy a databases between 32-bit and 64-bit systems or between big-endian and little-endian architectures.

These features make SQLite a popular choice as an Application File Format. SQLite database files are a recommended storage format by the US Library of Congress. Think of SQLite not as a replacement for Oracle but as a replacement for fopen() function in C language.

SQLite is a compact library. With all features enabled, the library size can be as less as 750KB, depending on the target platform and compiler optimization settings. There is a tradeoff b/w memory usage and speed. Nevertheless, performance is usually quite good even in low-memory environments. Depending on how it is used, SQLite can be faster than direct file system I/O.

1. *DIRECTORY STRUCTURE:*

**ecommerce\_project**

├── add\_products.py

├── app

│ ├── config.py

│ ├── forms.py

│ ├── \_\_init\_\_.py

│ ├── models.py

│ ├── static

│ │ └── img

│ ├── templates

│ │ ├── 2fa.html

│ │ ├── base.html

│ │ ├── cart.html

│ │ ├── checkout.html

│ │ ├── enable\_2fa.html

│ │ ├── index.html

│ │ ├── login.html

│ │ ├── product.html

│ │ ├── products.html

│ │ ├── register.html

│ │ ├── success.html

│ │ └── test\_checkout.html

│ ├── utils.py

│ └── views.py

├── requirements.txt

└── run.py

1. *IMPORTANT FILES WITH THEIR ROLE IN PROJECT:*

• **run.py**: The entry point of the application. It initializes the Flask app and runs it in debug mode.

• **requirements.txt**: Contains the list of required Python packages for the project.

• **add\_products.py**: A script to add products to the database.

• **App directory**: Contains the main application code.

• **config.py**: Handles the application's configuration settings.

• **forms.py**: Defines Flask-WTF forms used in the application.

• **\_\_init\_\_.py**: Initializes the application and its components

• **models.py**: Defines the SQLAlchemy database models.

• **Static**: Contains static files, such as images.

• **Templates**: Contains the Jinja2 templates for the application.

• **utils.py**: Contains utility functions for handling two-factor authentication

• **views.py**: Contains the route handlers for the application.

III. RESULTS

1. *BACKGROUND:*

As more and more transactions are done online, the risk of fraud and identity of threat has been increased. You are not alone in this. You may have the fear that someone can hack your account and steal the data or sensitive information during transactions. This may not only lead to financial loses, but also damage your trust in the website.

To address these concerns we propose implementing some security measures. At first we implemented secure password policies in which you have to create strong and unique passwords with the combination of alphanumeric and special characters, periodic password changes and no repeating of the last three passwords you have used. We also stored these passwords in a secure hashed format, so that they cannot be stolen easy.

In this we also added an extra layer of the security with 2-factor authentication in which you have to enter the one-time password using authenticator app. This occurs after entering the password in your login page. This looks like an extra step but it will reduce the risk of unauthorized access to your account.

Finally, we integrate a secure payment gateway that compiles with industry standards such as PCI-DSS. This will make sure that any sensitive information, such as credit card numbers will be securely handled and will not be a risk of interception by hackers.

1. *SCREENSHOTS*

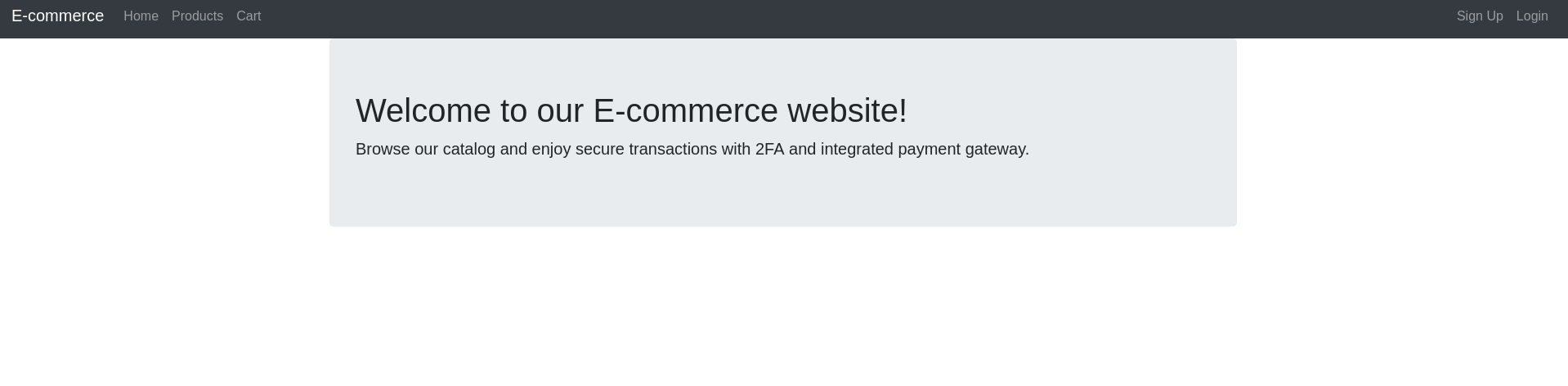


Figure 1: Landing Page

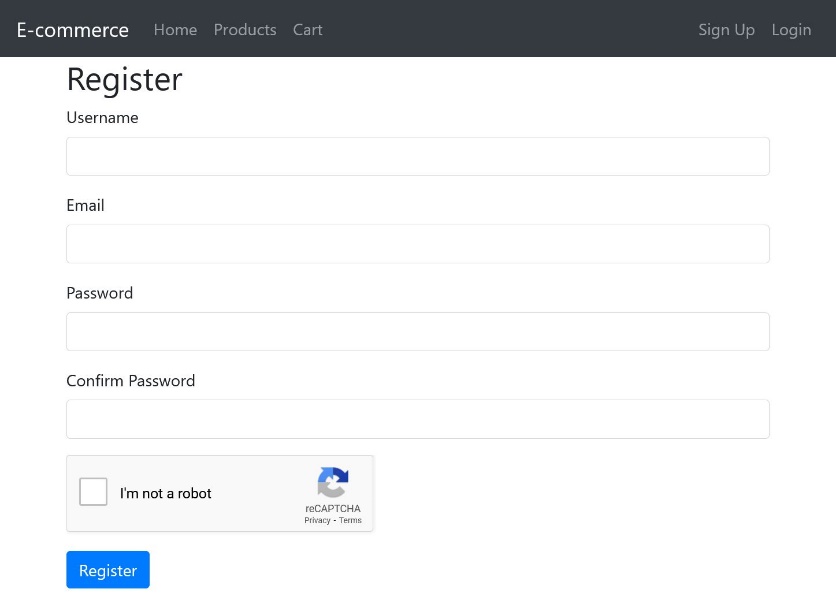


Figure 2: Sign up Page

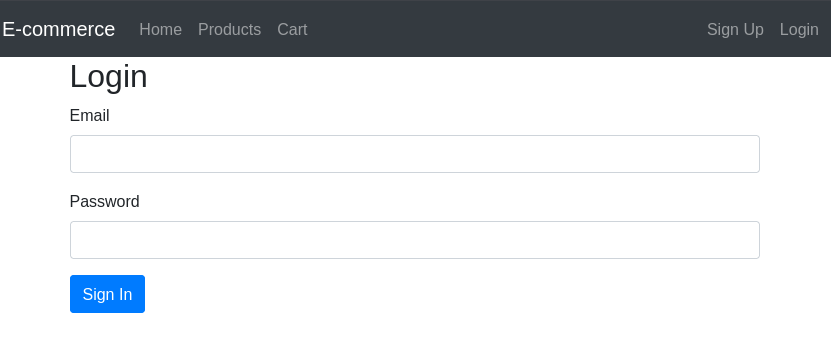


Figure3: Login Page

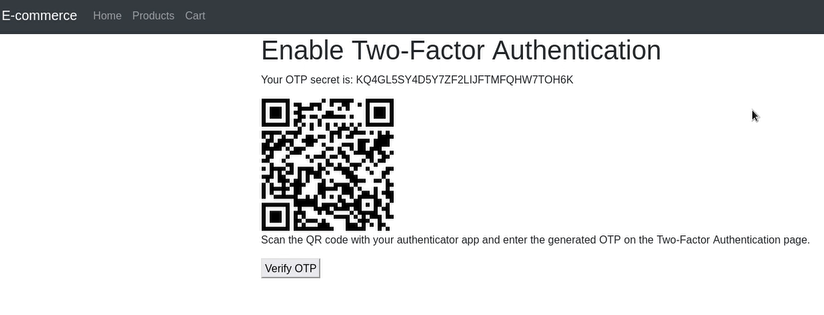


Figure 4: QR Authenticator Page

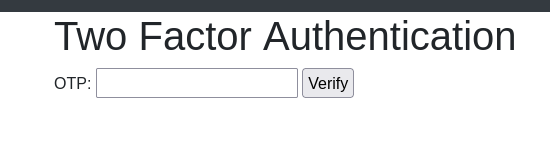


Figure 5: OTP Page

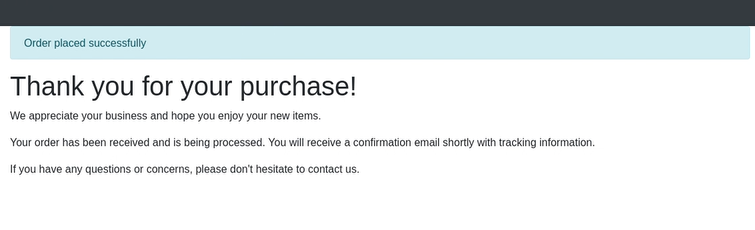


Figure 6: Success purchase Page

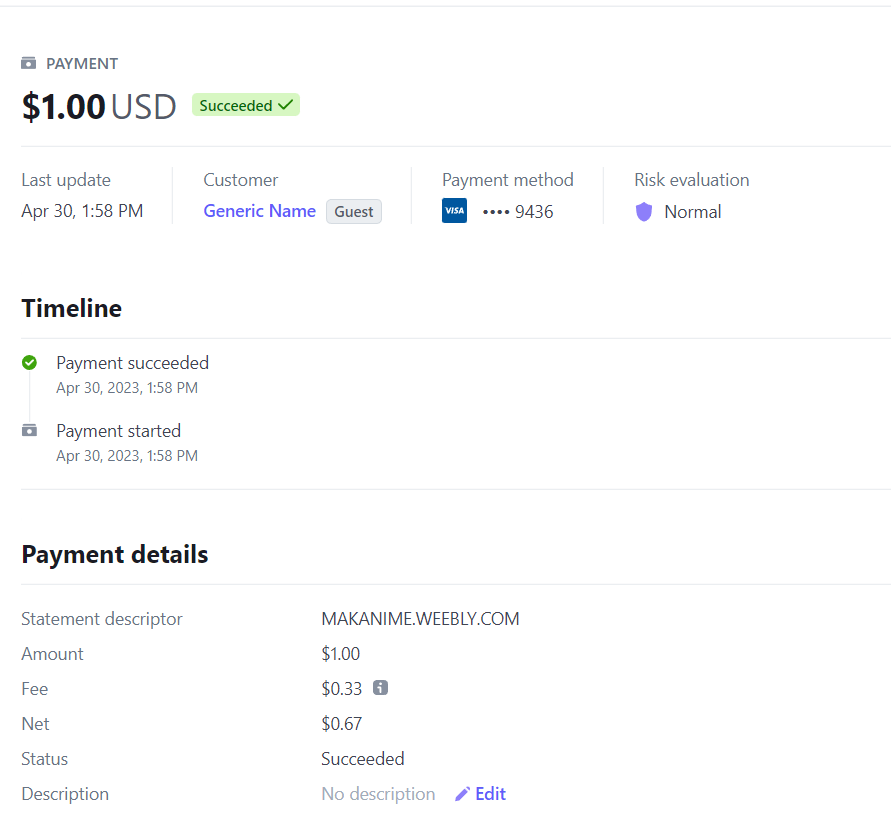


Figure 7: Stripe Payment Dashboard

IV. DISCUSSION AND CONCLUSION

The Internet has become a major resource in modern business, thus electronic shopping has gained significance not only from the entrepreneur’s but also from the customer’s point of view. For the entrepreneur, electronic shopping generates new business opportunities and for the customer, it makes comparative shopping possible. As per a survey, most consumers of online stores are impulsive and usually make a decision to stay on a site within the first few seconds. “Website design is like a shop interior. If the shop looks poor or like hundreds of other shops the customer is most likely to skip to the other site. Hence we have designed the project to provide the user with easy navigation, retrieval of data and necessary feedback as much as possible.

We generated report using OSWAP ZAP vulnerability testing tool. Below is the screenshot of the report. The testing report will be attached along with this document.

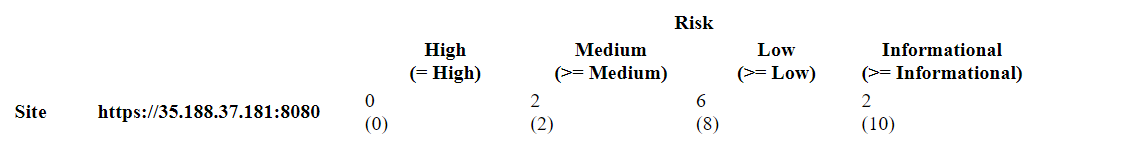


Figure 8: OWASP Report Sample

Below is the screenshot showing our instance is running using GCP.

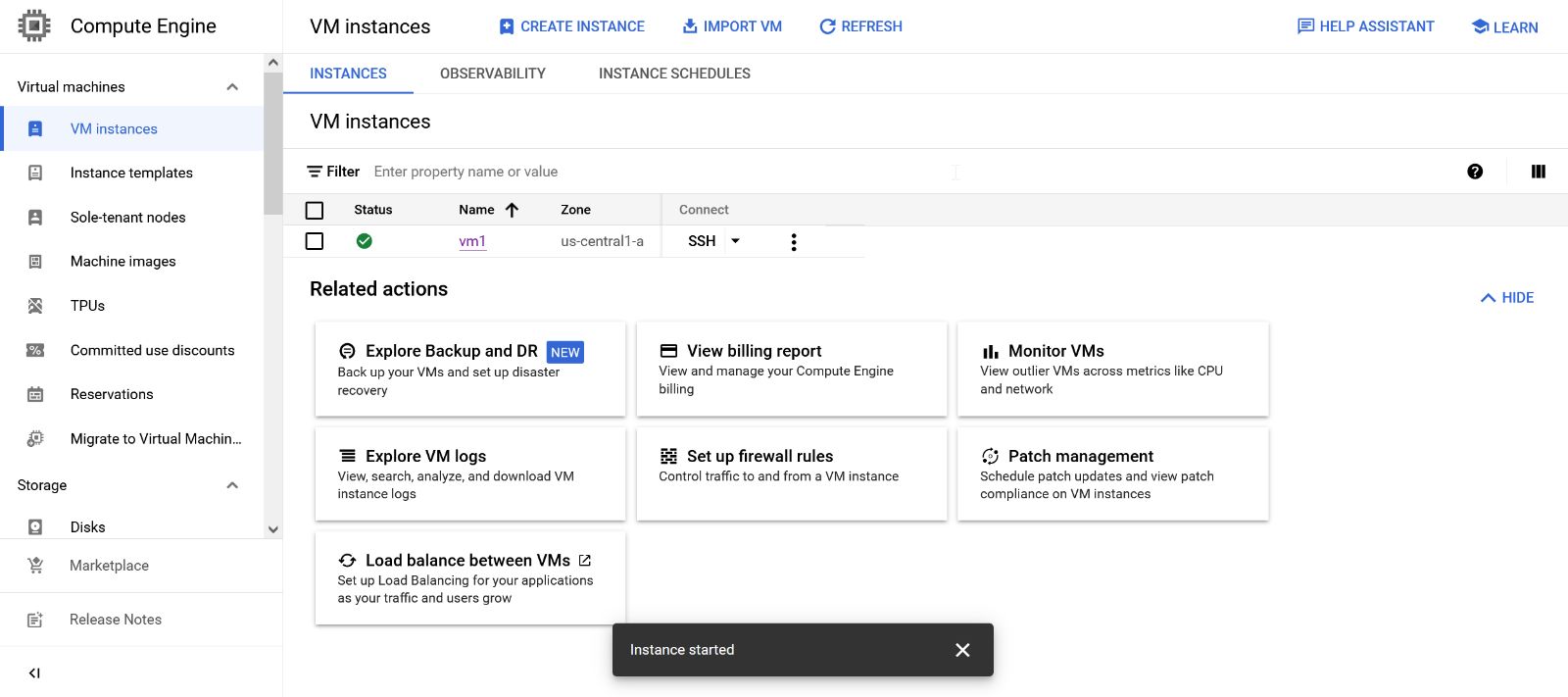


Figure 9: GCP Instance Running Page

V. REFERENCES

* <https://ethicalhackersacademy.com/blogs/ethical-hackers-academy/security-tools-for-e-commerce>
* <https://blog.rsisecurity.com/top-5-security-tools-in-e-commerce/>
* <https://ecommerceguide.com/guides/what-is-ecommerce/>
* <https://www.coursera.org/articles/ecommerce>
* <https://blog.miniorange.com/different-types-of-authentication-methods-for-security/>
* <https://www.fortinet.com/resources/cyberglossary/captcha>
* <https://www.fortinet.com/resources/cyberglossary/captcha>
* <https://softwarelab.org/what-is-a-captcha/>
* <https://www.forbes.com/advisor/business/software/best-payment-gateways/>
* <https://www.guru99.com/splunk-tutorial.html>
* <https://help.sharetribe.com/en/articles/1055989-configure-stripe-and-get-api-keys-for-a-marketplace>
* <https://medium.com/@mishraranjeet122/integrate-stripe-payment-with-a-card-in-python-e90989d39bca>