****

**Final Team Project**

**CSC 4022-470 Secure Software Engineering**

**FinApp**

**Financial Insights Web Application**

****

***Turing Tribe: Grant McPheeters, Mitchell Phelps, Aaron Garrison, Thea Banis,***

***Professor Rhodes***

***[insert your name]***

**Table of Contents**

[Project Overview 3](#_Toc149394921)

[Project Goals 3](#_Toc149394922)

[Project Features 3](#_Toc149394923)

[Target Audience 4](#_Toc149394924)

[Project Risks 5](#_Toc149394925)

[Design and Secure Architecture 6](#_Toc149394926)

# Project Overview

This project aims to create a user-friendly web application for financial insights, which will allow users to securely view and visualize their financial data from multiple different sources, including bank accounts, stocks, and liquid assets. The primary data source will be Plaid API, with possible integrations of other APIs for stocks and cryptocurrency data.

# Project Goals

Our primary objective is to offer users a centralized and user-friendly platform that empowers them to efficiently monitor their financial accounts and investments. We aim to provide a seamless and intuitive experience with data visualization. This involves employing interactive charts and graphs that allow users to gain valuable insights into their financial data with ease.

To ensure real-time accuracy and security, we integrate the Plaid API into our platform. Plaid’s robust infrastructure allows us to access financial data securely and reliably, while real-time updates guarantee that users have the most up-to-date information.

In our pursuit of enhancing the user experience, we also are considering the inclusion of additional APIs, specifically for stock market and cryptocurrency data. This expansion broadens the scope of information available to users, enabling them to make informed decisions regarding their investments.

Above all, we prioritize creating a user-friendly interface that fosters a smooth and enjoyable experience. This involves a design that simplifies navigation and ensures that users can access the information they need with ease. By focusing on these key elements, we aim to provide a comprehensive financial monitoring platform that meets the needs of our users while upholding the highest standards of data accuracy and security.

# Project Features

The project scope outlines the specific areas of focus and responsibilities:

* **User Authentication:** Users will have the option to create accounts and securely log in to the platform.
* **Financial Overview:** Users will be presented with an intuitive dashboard that provides an overview of their financial assets, including bank account balances, stock portfolios, and liquid assets.
* **Data Visualization:** This web application will consist of interactive charts and graphs to help users visualize their financial data effectively.
* **Plaid API Integration:** This application’s functionality will rely on the Plaid API to fetch and update user financial data.
* **Stock and Cryptocurrency Data:** Additional APIs will be considered for integrating stock market and cryptocurrency data, offering users a comprehensive financial snapshot.
* **User-Friendly Interface:** The user interface will be designed with ease of use in mind, ensuring a smooth user experience.

# Target Audience

The target audience for this application focuses on casual investors looking to monitor basic transactions and funds on all connected accounts and people who wish to see a nice roundup of all their financials in one application. This application is not anticipated to show advanced analytical data, so any advanced features would be done on the appropriate service.

Project Requirements

* **Functional Requirements**

Our functional requirements for this project are designed to ensure robust security, seamless user experience, and efficient data management. Firstly, user authentication is a critical aspect of our application design. Users will be required to authenticate themselves before gaining full access to the website. This process will be maintained throughout their interaction with the platform, ensuring that only authorized individuals can access their financial data.

Security will remain our top priority when dealing with the data exchange from Plaid and financial APIs. All data will be securely encrypted to protect it from unauthorized access, and that we have strict policies in place when it comes to deleting data securely that is no longer needed, thus minimizing potential and unnecessary exposure.

To enhance modularity and user experience, we will employ self-contained elements for each financial application connected by the user. These containers display all relevant data, such as invested funds, transactions, and graphic elements. Data between these containers will be carefully masked to prevent unauthorized data leakage.

To ensure that users have the most up-to-date information, we will prioritize polling historical data from Plaid and APIs whenever available, rather than relying on caching methods. This approach minimizes data exposure and guarantees the accuracy of the information.

Our graphical user interface (GUI) is designed to be compatible with various device resolutions and browsers, ensuring consistent and user-friendly experience for all users. We will prioritize HTTPS to secure data transmission and restrict the use of HTPP protocols, thus enhancing our platform’s overall security.

Finally, we will create modular graphics and charts for data display, allowing users to interact with their financial information in a visually engaging manner. These functional requirements combined will create a platform that not only prioritizes security, but also delivers a user-centric experience.

* **Non-Functional Requirements**

Our non-functional requirements encompass aspects of physical security and user privacy to enhance the overall user experience and safeguard sensitive financial information.

We will place a strong emphasis on physical security concerning balances and transactions. This includes controls to prevent individuals from viewing users’ financial information over their shoulder, a practice commonly known as ‘shoulder surfing’. To achieve this, we’ve designed the app to only display numeric values when prompted by the user. This approach adds an extra layer of security by concealing sensitive financial details unless explicitly authorized by the user.

By addressing these non-functional requirements, we not only enhance the overall security of the application, but also prioritize user privacy and data protection, contributing to a more robust and user-friendly financial monitoring experience.

Project Team

The project team will consist of the following roles sourced from Turing Tribe:

* Project Manager
* Systems Administrator (Backend Developer)
* Frontend Developer/ UX/UI Designer
* Quality Assurance Tester

The diversity of this team will bring a range of skills and knowledge necessary for the successful development and deployment of this financial insights’ web application.

# Project Risks

Preliminary Threat Assessment - While our primary focus is on providing valuable financial insights, we acknowledge the importance of security. Below is a list of our preliminary security considerations:

* **Data Security**

To ensure the security of our data, we will implement a comprehensive strategy. All stored data is encrypted, encompassing critical information like users’ usernames, passwords, and API keys. This encryption adds an extra layer of protection to sensitive information.

Additionally, we prioritize physical security by concealing information within the application unless explicitly requested by the user. This approach will minimize the exposure of data, reducing the risk of unauthorized access.

Furthermore, we enforce strict account permissions on our databases, differentiating between read and write access. This ensures that only authorized personnel can make changes to the data, enhancing overall data integrity.

To prevent potential vulnerabilities, we will also implement robust data sanitation procedures for all user input received by the program. This step is essential in guarding against SQL injection attacks, which are a common method used by malicious actors to manipulate databases.

Finally, we actively work towards mitigation strategies against cross-site scripting attacks, another prevalent threat vector. By staying vigilant and continuously enhancing our security measures, we aim to provide a safe and secure environment for our users’ data.

* **API Security**

When it comes to API security, we adhere to a multi-faceted approach to safeguard sensitive data and ensure smooth user interactions. Firstly, we emphasize the proper use of API keys, encompassing both private and public keys. This distinction is critical in controlling access to our APIs, as private keys grant high-level privileges and should be handled with the utmost care.

We prioritize transparency and user consent by seeking confirmation from users regarding the use of Plaid API. This step ensures that users are aware of and agree to the data exchange with the Plaid service, enhancing our trust and compliance with data protection regulations.

To prevent unauthorized access, we implement strict verification procedures for requests originating from the Plaid API. This verification process acts as a barrier against potential security breaches, guaranteeing only legitimate requests are processed.

In addition, we provide users with the option to save or not save API authentication with their accounts. This flexibility empowers users to control how their API credentials are managed, giving them a sense of ownership and control over their data. This approach not only enhances security but also respects user preferences. By combining these measures, we create a robust API security framework that ensures data integrity and user privacy.

* **User Authentication**

We place a strong emphasis on user security, starting with the creation of secure passwords. Users will be prompted to generate passwords that meet strict criteria, ensuring robust protection for their accounts. These passwords must meet a minimum of 12 characters in length, include at least one uppercase and one lowercase letter, incorporate at least one number, and feature at least one special character. This password policy is designed to thwart unauthorized access attempts and safeguard user data.

Following the password confirmation, we implement an additional layer of protection through multi-factor authentication. Users are prompted to confirm their identity through a secondary method, such as email or text verification. This multi-factor approach adds an extra barrier against potential breaches, making it significantly more challenging for unauthorized individuals to gain access to user accounts. By combining these security measures, we strive to create a safe and secure environment for our users, protecting their data from potential threats.

* **Updates**

When it comes to updates, our approach is rooted in adaptability. Throughout the planning and execution of our project, we maintain a proactive stance on addressing any security risks or flaws that may emerge. Our team is dedicated to staying current with the ever-changing threat landscape, which means that as new vulnerabilities are identified, we take immediate action to resolve them.

This commitment to ongoing updates allows us to continuously fortify our systems and applications against emerging threats. By staying agile and responsive in our security efforts, we uphold a strong defense against potential risks and maintain a high level of protection for our systems and users.

# Design and Secure Architecture

1. **Security Principles and Design**

* **Secure Design Principles**

Secure design principles are foundational to our approach in developing the FinApp application. These principles ensure we maintain a robust and comprehensive security posture.

Firstly, the principle of least privilege guides our approach. To achieve this each user will be granted only the minimum level of access required to perform their functions within the FinApp. This principle is explained in our Access Control Mechanisms section, where strict access control will be implemented to restrict unauthorized access.

We will also adhere to the concept of defense in depth. This involves the user of multiple security practices to safeguard the FinApp from various attacks and potential exploits. Our commitment to defense in depth is evident throughout the security measures detailed in this section, providing multiple layers of protection to ensure comprehensive security.

Lastly, we will integrate the principle of fail-safe defaults. This principle dictates that all users – including developers – are granted default system privileges with security in mind. For FinApp, this means that developers do not have plaintext access to users’ data by default, to minimize the risk of insider threats. They will be provided with specific functions solely for development purposes, ensuring that data security is maintained. For instance, front-end engineers will be given access to a test database by default, and not access to the actual database.

By adhering to these secure design principles, we create a secure environment for FinApp, where users’ data is protected, and potential vulnerabilities are mitigated effectively.

* **Threat Modeling**

In the development of FinApp, our primary focus is to deliver valuable financial insights while upholding the highest standards of security. Threat modeling plays a critical role in this process by helping us identify potential vulnerabilities and taking proactive steps to mitigate them.

Here are the key aspects of our threat modeling approach for this project:

1. **Data Security:** Given the financial nature of our application, data security is a paramount concern for us. To ensure the utmost protection, all stored information encompassing sensitive details like usernames, passwords, and API keys will be subject to robust encryption both at rest and during data transport. We will also prioritize the physical security for this information, concealing it within the application by default, unless the user specifically requests it in a different format.
2. **Access Control Mechanisms:** We will implement precise access control measures to guarantee that users and user groups have only the least privilege necessary. This ensures that users can access data strictly based on the actions they intend to perform within the application.

By addressing these key aspects through our threat modeling, we aim to create a financial insight application that not only delivers valuable data, but also upholds the highest standards of security.

* **Protection Mechanisms**

To fortify the security of FinApp and shield it from potential threats, we have established a range of protection mechanisms that align with standard secure design principles.

These mechanisms encompass the following key aspects:

1. **Privilege:** We adhere to the principle of least privilege, ensuring that users are granted only the minimum level of access necessary for their functions within the application. This approach guarantees that no user can access data beyond their specific needs and involves the careful configuration of administrative accounts to prevent unauthorized access.
2. **Defense in Depth:** Our security strategy involves a multi-layered approach, implementing security measures at various levels, ranging from the application layer to physical security. This comprehensive defense strategy safeguards FinApp against an array of potential attacks and exploits, thus reducing vulnerabilities.
3. **Access Control Mechanisms:** We define specific user groups with distinct privileges, and no single user has the capability to execute critical functions without the confirmation of another. This strict control ensures that users can only access data required for their intended actions.
4. **Encryption:** Strong encryption, such as AES, is employed for sensitive data during both transit and storage. This includes information like API keys, passwords, phone numbers, addresses, and names. At no point will any of this sensitive data be stored in plaintext, enhancing security and data protection.
5. **Threat Modeling:** We will conduct regular threat assessments and analysis to guide the continuous improvement of our security protocols and practices within the FinApp. This ongoing process ensures that our security measures remain robust and up to date to the evolving threats and vulnerabilities.

By incorporating these protection mechanisms, we aim to create a secure environment for FinApp. This will help mitigate potential risks and safeguard the sensitive data within.

1. **Access Control and Data Security**

* **Access Control Mechanisms**

Access control mechanisms are crucial in defining the privileges for specific users or groups within our FinApp application. These mechanisms ensure that users have only the data necessary for their intended actions, aligning with the principle of least privilege.

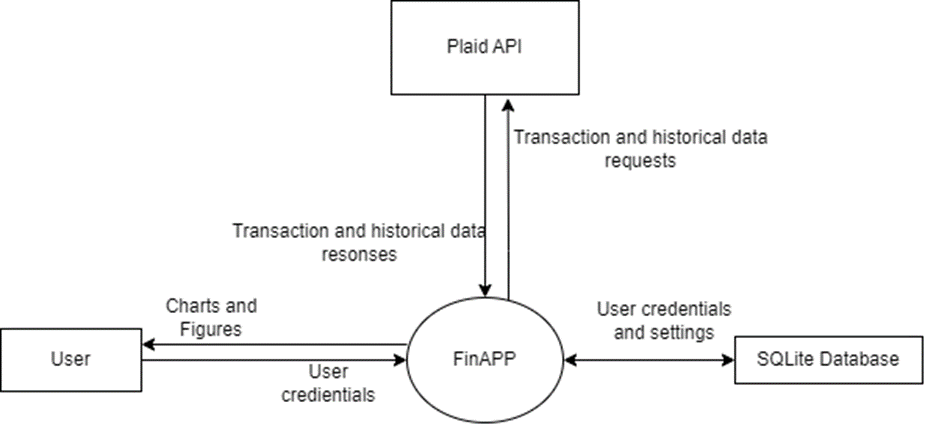
We categorized users into distinct groups, each with predefined privileges:

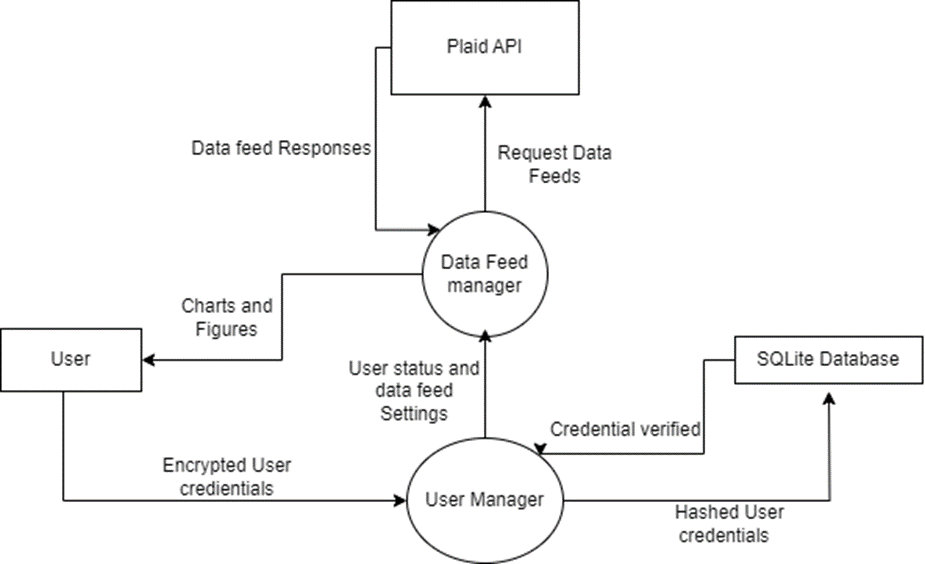
1. **Account Users:** These users will have access to their individual financial data, their own username, password, and API keys. It is critical that no user account can access the data of other users. This strict control of access preserves user privacy and data security.
2. **Test Users:** Test users are designated to access test data for evaluating the app’s functionality before creating an account. Like account users, test users are limited to accessing only the test data on the app, excluding any test API keys, passwords, or usernames related to test accounts. Additionally, the test user group serves as the primary development account for our developers, facilitating app testing and enhancement.
3. **Data Admin:** This group is granted access to the database, which includes hashed passwords and usernames. They will be strictly prohibited from accessing plain text passwords, API keys, or any financial data belonging to the users. This separation of access privileges ensures that sensitive user data remains protected even within the administrative role.

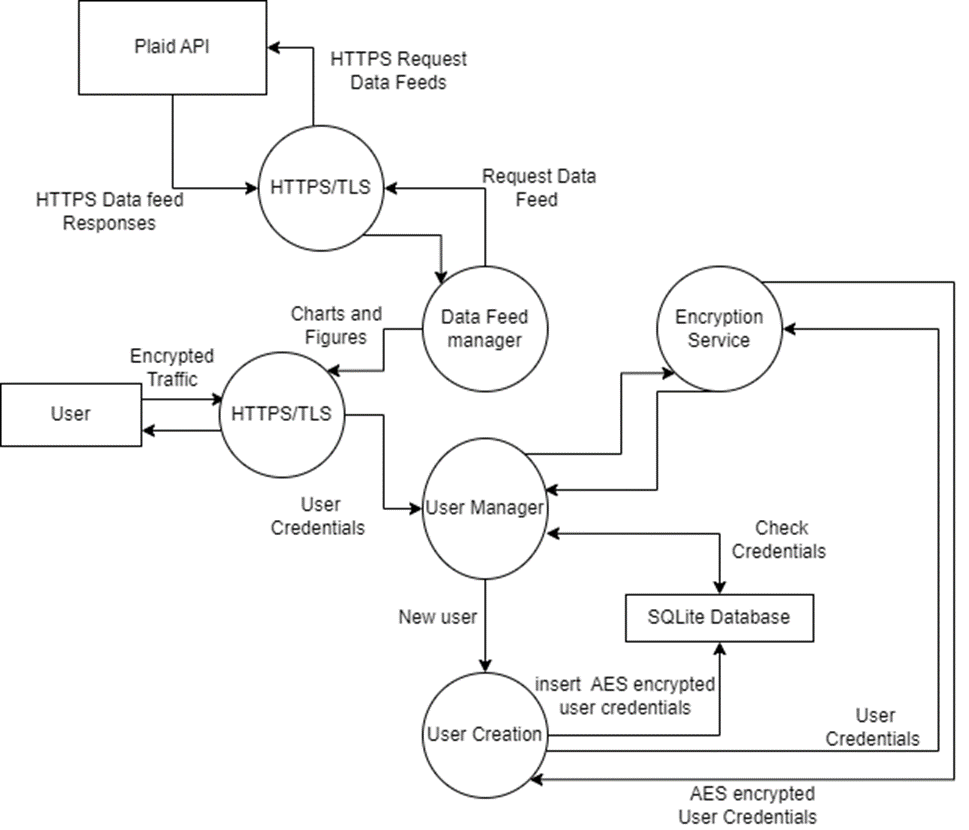
* **Data Flow and Storage Security**

Data flow and storage security are essential components of our strategy to maintain a secure environment for the application. Our approach involves carefully defining the data flow within the system, which is detailed in the attached images and documents available in the “DFD Documents and Images” folder within the repository.

Additionally, we prioritize the encryption of the database using Advanced Encryption Standard (AES). We will verify the identity of each user on every request made to the system. This combination of data flow management and storage security measures ensures that our application’s data remains confidential and secure.







* **Secure Communication Protocols**

Secure communication protocols are paramount for FinApp’s web application. To achieve this, we will rely on the HTTPS protocol, which will establish secure communication between our servers and the users’ browsers.

HTTPS ensures that communication parameters are established before any data transfer takes place. Although FinApp doesn’t facilitate direct financial transactions or money handling, it does contain exceptionally sensitive data. To fortify our security posture, we will implement robust security measures, including hashing and encryption as previously described.

These security measures serve as safeguards against vulnerabilities, such as man-in-the-middle attacks and spoofing, thus providing an additional layer of protection to ensure that FinApp remains resistant to unauthorized access and data breaches.

* **Encryption Strategies**

Encryption strategies are at the core of our data security measures, ensuring that information remains protected both during transit and while in storage.

Encryption serves as a fundamental pillar of our security approach. In accordance with industry standards, we will employ Advanced Encryption Standard (AES) encryption for all data within the FinApp. This includes sensitive information such as API keys, passwords, personal data, biometrics, routing numbers, transactions, and investments. By encrypting this data in storage, we add an additional layer of defense against potential breaches and unauthorized access.

Hashing is another integral component of our security strategy. It involves processing plaintext through a hashing algorithm to generate a seemingly random set of characters. On the server side of FinApp, we will implement hashing and salting techniques to verify passwords. Specifically, Django utilizes the PBKDF2 algorithm with a SHA256 hash for password stretching. Furthermore, data encryption within this framework incorporates a unique key known exclusively to the application server.

These encryption strategies ensure that data is shielded from potential threats, guaranteeing that sensitive user information remains confidential and secure throughout its lifecycle within the FinApp.

* **Error Handling and Logging**

Error handling and logging are integral aspects of our security strategy in FinApp. The primary focus here is to ensure that errors are managed in a manner that doesn’t compromise the security of users’ data.

In FinApp, our priority is to prevent errors from inadvertently granting users unauthorized access to data. For instance, we address the possibility of errors exposing another user’s API keys by associating API keys with specific user accounts, thus preventing any misuse of the keys.

In cases of fatal errors, user sessions will be promptly disconnected, and all financial data is deleted. Users will be notified about the occurrence of an error, and the incident will be logged for further analysis by system maintainers.

Logging holds a crucial role within FinApp. We emphasize logging specific events such as failed login attempts, changes in passwords and usernames, and the modification of API keys. This detailed logging approach equips our developers and administrators to identify potential security threats or breaches affecting specific user accounts. Additionally, we take measures to ensure that all logs avoid containing Personally Identifiable Information (PII) that could compromise user account security and privacy.

* **Secure API Design**

Secure API design is a fundamental consideration in the development of FinApp, particularly in our use of the Plaid API. This API facilitates connections to a range of banks and financial institutions, enhancing the capabilities of our application. The Plaid API authenticates users and verifies their identity through multiple functions. Among these functions, we utilize checks on the user’s address and phone number to ensure alignment with the bank account they are attempting to connect to. This secure API design ensures that the identity of our users is well-protected while enabling seamless and secure access to their financial data through the FinApp.

* **Third-Party Integration**

In the development of FinApp, we have incorporated several third-party libraries to enhance its functionality and security. These libraries primarily revolve around Django for the web service and for the security services we leverage either PyCyptodome or Cryptography. Given Django’s bundled SQLite, it is likely that this built-in database system will be utilized to efficiently manage and store data for this application. This strategic use of third-party integration enhances FinApp’s capabilities while maintaining robust security measures.

1. **Development and Quality Assurance**

* **Scalability and Performance**

Django will be our chosen framework for FinApp, which is renowned for its scalability and adaptability in handling varying workloads, whether they are large or smaller in enterprise scale. As the demands of the application will fluctuate, Django offers the flexibility to transition application states to different technologies as needed. These customizations encompass various aspects such as database management, session handling, and file storage. This will ensure that the application can seamlessly adjust to evolving performance requirements, ultimately optimizing scalability and the application’s efficiency.

* **Documentation**

Our comprehensive documentation for FinApp encompasses key aspects of the application’s design and security protocols.

Firstly, we have defined three distinct user groups: Account Users, Test Users, and Data Admins, each with specific privileges and defaults as outlined in our Access Control Mechanisms. This documentation will serve as a reference point for the development team, ensuring that the security measures are correctly implemented for each user group.

Regarding data security, we employ robust encryption using Advanced Encryption Standard (AES) for sensitive information such as passwords, API keys, phone numbers, and addresses. Password verification is handled through secure hashing and salting methods, which will be thoroughly documented, guaranteeing that the server remains unaware of users’ sensitive data.

In terms of error handling, our approach is to address errors securely and prevent vulnerabilities. Critical errors will be documented, including the immediate disconnection of the user and the secure deletion of all financial information. This documentation ensures that the application remains resilient to security threats, and developers have clear instructions on how to handle errors.

Logging is a vital component of our security strategy, covering critical events such as login attempts, changes in passwords and usernames, and modifications to API keys. This documentation will specify what events are logged and how the system will respond to them. This approach will allow us to monitor and respond to security incidents effectively, as explained in our Incident Response Plan section.

Finally, in our secure API design we will make full use of the Plaid APIs built-in verification and authentication methods. The documentation will provide a step-by-step guide on how these methods are integrated to enhance the security of our API calls. This will ensure that user data remains well-protected, and that the integration is well-documented.

Through this comprehensive documentation, we will establish a clear and robust framework for the security and functionality of the FinApp.

* **Peer Reviews**

Peer reviews are an integral part of our development process. During this phase, we will conduct these reviews through pull requests for all feature branches created. These reviews will focus not only on the code itself, but also on the implementation of security mechanisms. We will ensure that the code adheres to industry standards and meets all necessary security requirements. This rigorous peer review process, documented in our workflow, ensures the quality and integrity of our codebase as we further build onto the application.

* **Iterative Design**

As the Turing Teams gain insights into new, secure implementations and strategies, this document remains open to modification to incorporate more appropriate security protocols. We encourage ongoing discussions with stakeholders to uncover changing requirements that influence our secure design. This iterative approach, detailed in the documentation, allows us to continuously improve and adapt our design to meet evolving needs and security challenges.

We also emphasize continuous threat assessment, which may lead to protocol adjustments as we enhance our understanding of optimizing the use of the Plaid API and further develop the FinApp. In terms of feature development, we will follow a structured approach where smaller sprints are employed as individuals are assigned modules to build. These modules will serve as the building blocks to creating our larger system, which will undergo rigorous unit and integration testing to ensure their reliability and security.

* **Review with Stakeholders**

We prioritize ongoing engagement with our stakeholders in the FinApp development process. We have established a monthly meeting schedule to facilitate discussions on potential changes to functionality and the introduction of new features. These interactions also involve consultations with security professionals to enable comprehensive security analysis throughout the development cycle.

Our approach of engaging with stakeholders and security experts early in the process is designed to identify and address potential vulnerabilities at an early stage. This proactive approach ensures that any issues are promptly resolved, ultimately leading to a more secure release of FinApp to the public.

* **Design Validation**

Design validation is a crucial part of our development process, and we conduct it through a series of unit and integration testing. These unit tests are conducted to assess the functionality of various components:

1. **Encryption Unit Test:** This verifies the effectiveness of encryption modules in both encrypting and decrypting data.
2. **Database Unit Test:** This focuses on ensuring the successful operations of the database, including adding, modifying, and deleting data within it.
3. **User Creation Unit Test:** This confirms that users can be created successfully and can log in without issues.
4. **API Unit Test:** We can run tests on the API functions, including tests for successful logins, queries, and logouts.
5. **Visual Chart Unit Test:** This assesses the creation of visual charts using mock data and tests the functionality of each offered chart type.

**Integration Testing** will involve evaluating the interaction of various modules within the FinApp:

1. **Finance Module Unit Test:** We will create a simulated module used across each individual application with FinApp, covering aspects such as balances, charts, transactions, and data sourced from APIs. This will ensure that these modules integrate effectively and function as intended.

Through this rigorous testing approach, we aim to validate the design and functionality of FinApp, thus ensuring that all components work seamlessly and securely together.

1. **User Training and Incident Response**

* **User Training**

User training is an important aspect our project to ensure that users, administrators, and developers can effectively use the new security features that we wish to implement. The aim is to make the transition to the enhanced security measures as user-friendly as possible.

This training will encompass the following key components:

1. **User Awareness Workshops:** These workshops are going to be thoughtfully designed to be engaging and inclusive for all our users. We will use a variety of teaching methods, including interactive demonstrations, practical exercises, and user-friendly materials to cater to various learning styles. The primary aim is to empower users with the knowledge and skills required to make the most of our application while also safeguarding their sensitive financial information.
   1. We understand that the ease of using our application is a top priority to our users. In these workshops, we will provide a step-by-step guidance on how to navigate the application seamlessly. Users will learn about the features and functions that enhance their financial insights.
   2. Users will be equipped with the ability to identify and combat common phishing attacks by malicious actors. We’ll illustrate real-world examples of tactics used by cybercriminals and educate users on how to spot these threats and take appropriate action.
   3. We’ll also share insights into what data breaches that have occurred in financial institutions due to human error. These examples will serve as cautionary tales, emphasizing the importance of data management.
2. **Administrator Training:** Administrators play a vital role in managing user accounts, permissions, and overall system security. Specialized training sessions will be organized to equip administrators with the knowledge and tools they need to oversee these security measures effectively.
3. **Developer Guidance:** Developers who are responsible for maintaining and updating the application will receive guidance on the new security features. This includes understanding the code changes, configurations, and best practices to ensure the application remains secure and functional.

* **Incident Response**

Security incidents can arise during the operation of the system, and a well-defined incident response plan is essential to handle them effectively.

This plan will encompass the following key aspects:

1. **Incident Identification and Classification:** We will have monitoring systems in place to detect and identify security incidents and unexpected behaviors. This will include real-time monitoring and automated alerts for suspicious activities. These incidents will be classified based on severity to determine the urgency and level of response needed.
2. **Response Team:** A dedicated incident response team will be established, consisting of on-call IT security professionals.
3. **Containment:** Once an incident has been reported, the response team will follow predefined procedures to isolate the affected systems.
4. **Eradication:** After the affected systems are contained, the focus will shift to patching the root cause of the incident.
5. **Recovery:** We will have documented recovery procedures to restore the system to a nominal state, whether this be restoring the original state or reestablishing the resources to a hot/cold site available. In this process we will also be verifying the integrity of our data and systems.
6. **Communication:** Maintaining clear and concise communication is vital during security incidents. This includes having a well-defined plan of action for promptly and transparently informing relevant stakeholders and users. By doing so, we not only address the incident effectively, but also uphold our trust and reputation with the public.
7. **Documentation:** Every step of the incident response will be documented for review.
8. **Post-Incident Review:** After an incident is resolved, a post-incident review will be conducted to understand what happened, how it was handled, and how we can prevent similar incidents in the future.

By incorporating user training and incident response planning into our project, we not only enhance the security and usability of the financial insights’ web application, but also ensure that users are well-prepared to interact with the application’s security features and address security incidents when they occur.