# CS 305 Project One

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **November 17, 2024** | **Robert DiMaio** | **Initial submission** |
| **2.0** | **November 22, 2024** | **Robert DiMaio** | **Resubmission; removed/edited some content from dependency check for lower Turnitin Score** |

## Client



**Developer**

Robert DiMaio

1. **Interpreting Client Needs**

Our client, Artemis Financial, is a consulting company that develops individualized financial plans for its customers. Because they are a financial company, secure communications is paramount. Financial institutions and any communications and transmissions dealing with money are arguably the most targeted and have some of the highest risk of any software security considerations. The customer needs to know they are choosing a company that takes security as seriously as they do with their own money.

While not explicitly stated within the scenario, it can be safely assumed that Artemis Financial handles international transactions, especially since our company, Global Rain, works with different companies around the world. Whether they handle international transactions or not, there are various governmental restrictions on secure communications that need to be considered. In the US: “The [Gramm-Leach-Bliley Act](https://www.ftc.gov/legal-library/browse/statutes/gramm-leach-bliley-act) requires financial institutions – companies that offer consumers financial products or services like loans, financial or investment advice, or insurance – to explain their information-sharing practices to their customers and to safeguard sensitive data.” (Federal Trade Commission, n.d.) If operating in the EU, then the General Data Protection Regulation applies. The GDPR mandates strict rules on data handling, encryption, and transmission to protect user privacy. (GDPR Info, n.d.)

There are currently, and realistically will always be, external threats on any kind of financial institution. The incentive for hackers to try to gain access to money is too great to be stopped. Though we will likely never stop the attackers from trying, we can do everything in our power to prevent them from gaining any kind of unauthorized access to our users’ information and money. Some examples of external threats are phishing, ransomware, and denial-of-service attacks. All of these can be at the least, disruptive if successful, if not devastating to the company and users alike.

Technology is ever-evolving, and we must consider various modernization requirements, such as the role of open-source libraries and evolving web application technologies. With new features and software updates comes more open-source libraries and evolving web application technologies. While they can be an asset, they are also not without their risks. Any new code that gets integrated into our operations needs to be thoroughly vetted and checked for vulnerabilities and dependencies.

**2. Areas of Security**

The areas of security outlined in the Vulnerability Assessment Process Flow chart are all critical elements to be aware of for many software applications, however, for this particular codebase, some stand out more than others. The areas of security that are most applicable for this application are Input Validation (Secure Input and Representations), Code Error (Secure Error Handling), Code Quality (Secure Coding Practices/Patterns), and Encapsulation (Secure Data Structures).

Any piece of software that may accept any kind of input needs thorough and secure input validation. Without proper input validation, our code can be susceptible to undesired input, whether malicious or by accident, that can cause unintended behavior within the program and can cause critical errors. Any software can and will have errors, and those need to be handled securely and prevent the application from crashing. Secure coding practices and patterns are paramount in any sort of software development and this application is no exception. Ensuring proper encapsulation with secure data structures is another way to keep the program running as intended. Arguably all of these areas of security fall under code quality and secure coding practices/patterns, and much of that comes from various forms of input validation.

**3. Manual Review**

**Input Validation**

In the CRUDController.java file the @RequestParam parameter name is directly accepted without any validation or sanitization. This can lead to unintended behavior within the program.

In the customer.java file the deposit() method directly modifies account\_balance without validating the input. This could lead to negative or invalid deposits if called with inappropriate values.

**Code Error**

The CRUDController.java code doesn’t include any error handling. If DocData encounters an error (e.g., NullPointerException or parsing issues), it could lead to runtime errors.

The e.printStackTrace() method within DocData.java prints the stack trace to the standard error, which should invoke more advanced error handling.

Code Quality

“In [Java](https://www.theserverside.com/definition/Java), all classes, interfaces and enums are expected to use Pascal case.” (TheServerSide, n.d.)

Both customer.java and myDateTime.java do not adhere to these standards and should be renamed to comply and maintain readability and consistency.

Within customer.java the account\_number and account\_balance fields have package-private (default) and private access modifiers. This may be fine if the class is only used internally, but it can lead to issues if unintended access is possible from other classes within the same package.

In DocData.java the database connection URL, username, and password are hardcoded directly in the read\_document() method. This practice is unsafe as it can lead to accidental exposure of sensitive information.

Within GreetingController.java the name parameter is passed directly from the URL query string into String.format(). This could lead to potential security issues if name contains unexpected input that could be processed in an unsafe way when displayed, which falls under code quality as well as input validation.

**Encapsulation**

In myDateTime.java the fields mySecond, myMinute, and myHour are package-private (default) and are accessible by other classes in the same package which can break encapsulation and can lead to unexpected behavior if these fields are modified directly.

**4. Static Testing**

We used the Maven Dependency Check to generate a report analyzing our codebase for known published vulnerabilities. Included below are all the vulnerabilities that were found via the dependency check and the associated CVE codes. Most of these vulnerabilities can be mitigated using rigorous input validation as well as making sure we are using the most up to date version of any of the dependencies. Many security issues that were present in older versions have been addressed in the updated versions.

**bcprov-jdk15on-1.46.jar**

**Published Vulnerabilities**

[**CVE-2016-1000338**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000338)

[**CVE-2016-1000342**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000342)

[**CVE-2016-1000343**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000343)

**CVE-2024-29857**

[**CVE-2016-1000344**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000344)

[**CVE-2016-1000352**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000352)

**CVE-2024-30171**

[**CVE-2016-1000341**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000341)

[**CVE-2016-1000345**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000345)

[**CVE-2017-13098**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-13098)

[**CVE-2020-15522**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-15522)

**CVE-2020-0187**

[**CVE-2023-33202**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-33202)

**CVE-2020-26939**

**CVE-2023-33201**

[**CVE-2016-1000339**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000339)

[**CVE-2015-7940**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2015-7940)

[**CVE-2018-5382**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2018-5382)

[**CVE-2013-1624**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-1624)

[**CVE-2016-1000346**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000346)

**CVE-2015-6644**

**hibernate-validator-6.0.18.Final.jar**

**Published Vulnerabilities**

**CVE-2023-1932**

[**CVE-2020-10693**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-10693)

**jackson-databind-2.10.2.jar**

**Published Vulnerabilities**

[**CVE-2020-25649**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-25649)

[**CVE-2020-36518**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-36518)

[**CVE-2021-46877**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-46877)

[**CVE-2022-42003**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42003)

[**CVE-2022-42004**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42004)

[**CVE-2023-35116**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-35116)

**log4j-api-2.12.1.jar**

**Published Vulnerabilities**

[**CVE-2020-9488**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9488)

**logback-core-1.2.3.jar**

**Published Vulnerabilities**

[**CVE-2023-6378**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-6378)

[**CVE-2021-42550**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-42550)

**snakeyaml-1.25.jar**

**Published Vulnerabilities**

[**CVE-2022-1471**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-1471)

[**CVE-2017-18640**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-18640)

[**CVE-2022-25857**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-25857)

[**CVE-2022-38749**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38749)

[**CVE-2022-38751**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38751)

[**CVE-2022-38752**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38752)

[**CVE-2022-41854**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-41854)

[**CVE-2022-38750**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38750)

**spring-boot-2.2.4.RELEASE.jar**

**Published Vulnerabilities**

[**CVE-2023-20873**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20873)

[**CVE-2022-27772**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-27772)

[**CVE-2023-20883**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20883)

**spring-boot-starter-web-2.2.4.RELEASE.jar**

**Published Vulnerabilities**

[**CVE-2023-20873**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20873)

[**CVE-2022-27772**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-27772)

[**CVE-2023-20883**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20883)

**spring-core-5.2.3.RELEASE.jar**

**Published Vulnerabilities**

[**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965)

[**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118)

[**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421)

[**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950)

[**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971)

[**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861)

[**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863)

[**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968)

[**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970)

[**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060)

[**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096)

**spring-expression-5.2.3.RELEASE.jar**

**Published Vulnerabilities**

[**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965)

[**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118)

[**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421)

[**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950)

[**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971)

[**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861)

[**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863)

**CVE-2024-38808**

[**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968)

[**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970)

[**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060)

[**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096)

**spring-web-5.2.3.RELEASE.jar**

**Published Vulnerabilities**

[**CVE-2016-1000027**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000027)

[**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965)

**CVE-2024-38809**

**CVE-2024-22243**

**CVE-2024-22262**

[**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118)

[**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421)

[**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950)

[**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971)

[**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861)

[**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863)

[**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968)

[**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970)

[**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060)

[**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096)

**spring-webmvc-5.2.3.RELEASE.jar**

**Published Vulnerabilities**

[**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965)

**CVE-2024-38816**

[**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118)

[**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421)

[**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950)

[**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971)

[**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861)

[**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863)

[**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968)

[**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970)

[**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060)

[**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096)

**tomcat-embed-core-9.0.30.jar**

**Published Vulnerabilities**

[**CVE-2020-1938**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1938)

[**CVE-2020-11996**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996)

[**CVE-2020-13934**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934)

[**CVE-2020-13935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13935)

[**CVE-2020-17527**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-17527)

[**CVE-2021-25122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25122)

[**CVE-2021-41079**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-41079)

[**CVE-2022-29885**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-29885)

[**CVE-2022-42252**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42252)

[**CVE-2023-44487**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-44487)

[**CVE-2023-46589**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-46589)

[**CVE-2020-9484**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9484)

[**CVE-2021-25329**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25329)

[**CVE-2021-30640**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-30640)

[**CVE-2022-34305**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-34305)

[**CVE-2023-41080**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-41080)

[**CVE-2021-24122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-24122)

[**CVE-2021-33037**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-33037)

[**CVE-2023-42795**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-42795)

[**CVE-2023-45648**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-45648)

[**CVE-2024-21733**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2024-21733)

[**CVE-2019-17569**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569)

[**CVE-2020-1935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1935)

[**CVE-2020-13943**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13943)

[**CVE-2023-28708**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-28708)

[**CVE-2021-43980**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-43980)

**tomcat-embed-websocket-9.0.30.jar**

**Published Vulnerabilities**

[**CVE-2020-1938**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1938)

[**CVE-2020-8022**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-8022)

[**CVE-2020-11996**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996)

[**CVE-2020-13934**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934)

[**CVE-2020-13935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13935)

[**CVE-2020-17527**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-17527)

[**CVE-2021-25122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25122)

[**CVE-2021-41079**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-41079)

[**CVE-2022-29885**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-29885)

[**CVE-2022-42252**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42252)

[**CVE-2023-44487**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-44487)

[**CVE-2023-46589**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-46589)

[**CVE-2020-9484**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9484)

[**CVE-2021-25329**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25329)

[**CVE-2021-30640**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-30640)

[**CVE-2022-34305**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-34305)

[**CVE-2023-41080**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-41080)

[**CVE-2021-24122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-24122)

[**CVE-2021-33037**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-33037)

[**CVE-2023-42795**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-42795)

[**CVE-2023-45648**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-45648)

[**CVE-2024-21733**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2024-21733)

[**CVE-2019-17569**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569)

[**CVE-2020-1935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1935)

[**CVE-2020-13943**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13943)

[**CVE-2023-28708**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-28708)

[**CVE-2021-43980**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-43980)

**5. Mitigation Plan**

In this project, we performed a manual code check, as well as ran a dependency check for known vulnerabilities in the code. Using the information we have gleaned from these two processes, and following the Vulnerability Assessment Process Flow chart, we can then take steps to mitigate these security risks. For this analysis, our primary focus would be on the following areas of security: input validation (secure input and data representation), error handling (secure error management), code quality (adherence to secure coding practices), and encapsulation (secure data structures). Most vulnerabilities seem to stem from inadequate input validation, which is also an essential aspect of secure coding practices. Our current codebase has minimal input validation, and standard validation methods are often insufficient to protect against these well-known vulnerabilities. More rigorous input validation is necessary, following the principle of whitelisting—accepting only desired input and rejecting everything else. This approach is generally more effective than blacklisting, which only rejects known harmful inputs, as new exploits frequently emerge. However, combining both techniques can enhance security by reinforcing whitelisting with targeted defenses against specific threats. Additionally, encapsulation can benefit from robust input validation to prevent improper creation or access to data structures when attributes are not adequately checked. A comprehensive error-handling system is crucial to prevent unexpected behavior when inputs do not align with the program’s specifications.

**References**

Federal Trade Commission. (n.d.). *Gramm-Leach-Bliley Act*. <https://www.ftc.gov/business-guidance/privacy-security/gramm-leach-bliley-act>

GDPR Info. (n.d.). *General Data Protection Regulation (GDPR)*. <https://gdpr-info.eu/>

TheServerSide. (n.d.). *What is Pascal case? Definition from WhatIs.com*. TheServerSide. https://www.theserverside.com/definition/Pascal-case