1. ***What software methodology did I used?***

***Why did I choose this methodology?***

Software methodology systematic approach to the development, maintenance, and enhancement of software products is referred to as software methodology. It entails the use of procedures, techniques, and tools in order to assure the efficient and effective delivery of high-quality software that satisfies the demands of the consumers. A software methodology offers a framework for software development and aids in the management of the complexity of the software development life cycle. Agile, Waterfall, DevOps, and Scrum are some common software techniques, each with its own set of concepts and practices. The choice of a software methodology is determined by the unique demands of a project as well as the preferences of the development team.

**Methodologies:**

Traditional methodologies:

1. Waterfall methodology.
2. Alterative methodology.
3. spiral methodology

Advanced methodologies:

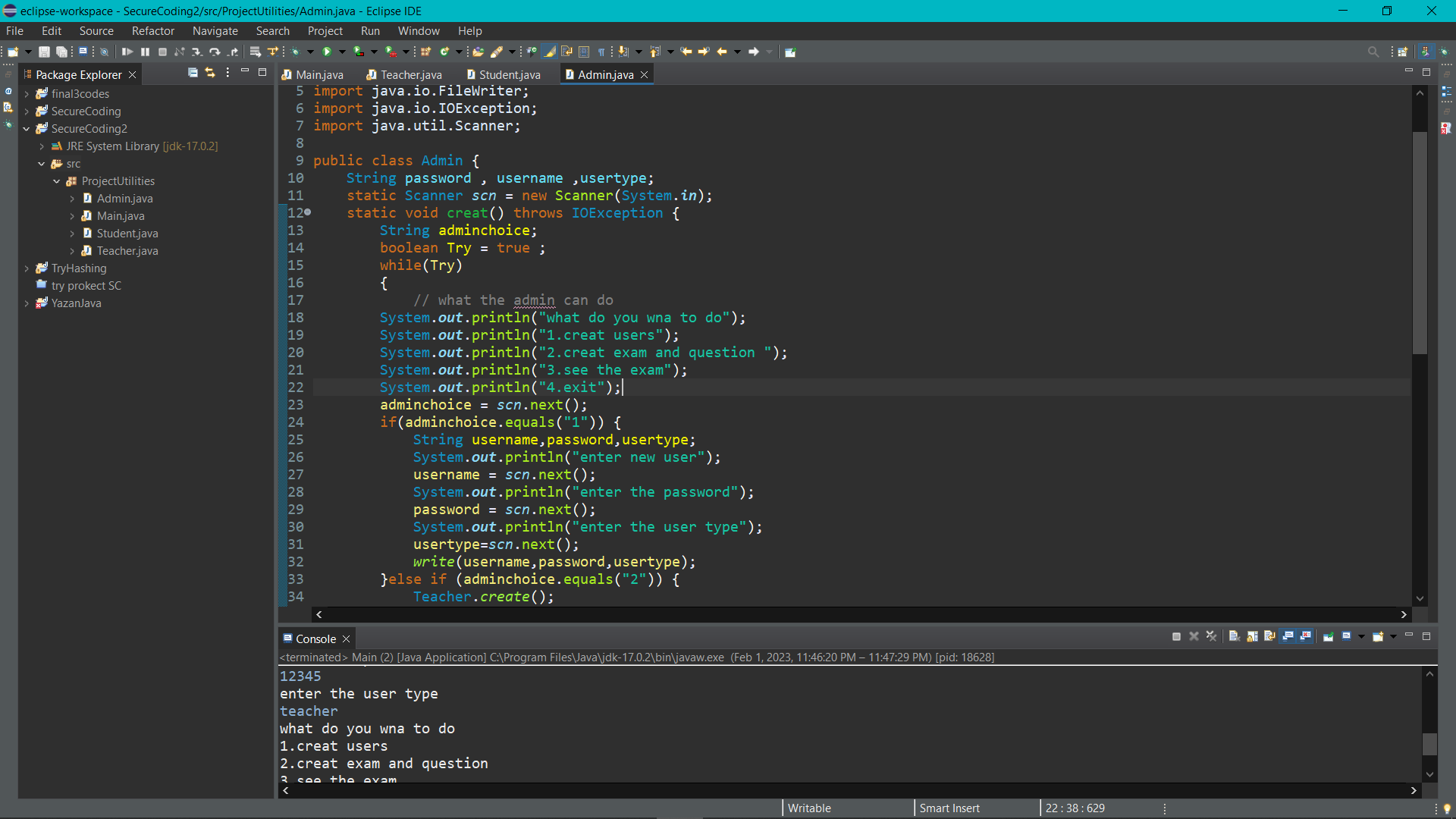
1. Agile development.
2. Hybrid situation.

Why I used agile methodology? I used agile methodology because it stresses flexibility, cooperation, and continual improvement, agile methodology is an excellent technique for Java development projects. Agile allows development teams to adapt fast to changes in requirements, resolve possible issues early on, and guarantee that the ultimate product fulfills the demands of stakeholders. Agile is iterative structure enables teams to release functioning software on a frequent basis and get feedback, resulting in a more efficient and effective development process. Furthermore, the tight cooperation promoted by Agile between developers, testers, and stakeholders helps to guarantee that everyone is on the same page and working toward a single objective. You may benefit from a more flexible, collaborative, and efficient software development process if you use agile technique for Java development.

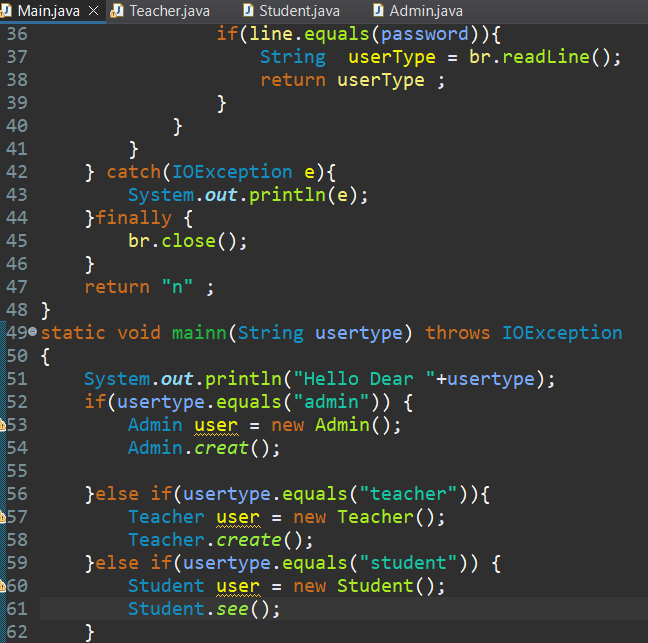
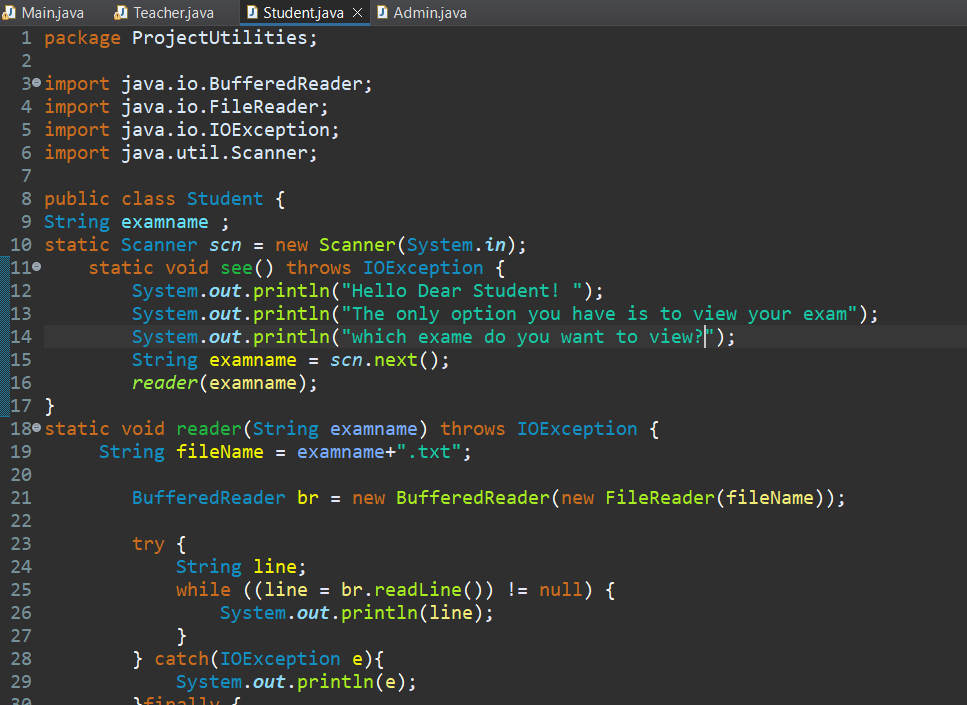
1. ***Listing security principles that were developed in my application.***

***Explaining the benefits for each principle*.**

**Least privilege:** is the notion of allowing only the least amount of access required to complete a task. This helps to restrict an attacker's potential attack surface by limiting the locations he or she can target and the amount of damage they can cause if successful. Furthermore, you can see that in my code every one of the following options like (student, admin, teacher) have a specific job to do, so that my work could be done in a professional way, and without any mistakes also due to the use of the classes I did. However, you can check my work in the following picture.



**Keep it simple:** highlights the necessity of keeping security design and implementation simple. Simple systems are less likely to contain hidden vulnerabilities and are easier to secure and maintain. Organizations may decrease the risk of misconfigurations and improve the efficiency of their security posture by keeping security measures simple. Furthermore, I kept my code simple by no exceeding more than 65 line in each class that were created, and the naming convention is also implemented in the code so anyone who is willing to edit or work in the code will not face any problem. However, I will put some picture in the following slide to show what I am talking about.

1. ***Listing two quality principles that were illustrated in my application****.*

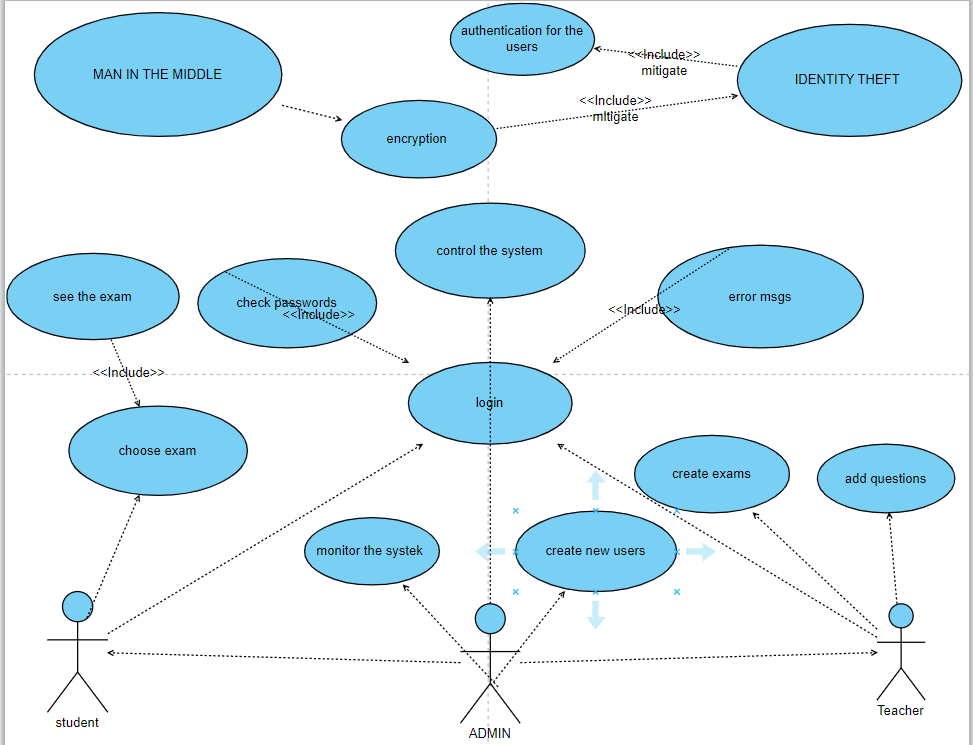
***Explaining the benefits for each principle. (with code)***

**Readability and capability:** refers to a product or service's ability to be easily understood and used by its target audience. Increased user happiness, lower training costs, and higher productivity are all advantages of readability and capability.

**Flexibility and reusability:** entails the capacity to modify a product or service to satisfy changing demands and be employed in a variety of situations. Reduced costs, higher productivity, and enhanced capacity to satisfy client demands are all advantages of flexibility and reusability. Products and services that are flexible and reusable are also easier to maintain and update, which may lead to longer lives and higher customer satisfaction. Organizations may guarantee that their goods and services stay relevant and valuable to consumers over time by stressing flexibility and reusability.

4. ***List the assets of the application.***

1. Grades
2. Exams
3. Users password
4. User information
5. ***Draw the use case diagram and the misuse case diagram for the application***

******

***6. List two of the misuse cases and list the countermeasures that I used.Justify the countermeasures chosen.***

**Identity fraud/theft:** To protect against identity fraud or theft, countermeasures such as two-factor authentication or providing a one-time password through SMS are recommended to validate the user's identity and secure critical information.

**Man in the middle:** To combat man-in-the-middle threats, use countermeasures such as encrypted VPNs for online communications, securing endpoint devices to prevent malware attacks, and implementing multi-factor authentication, which requires more than a username and password for access.

***7. Evaluation of the used countermeasures in the application, if they are enough or not.***

**Identity Fraud/Theft**: Two-factor authentication and the usage of one-time passwords delivered through SMS are excellent deterrents against identity fraud and theft. By requiring the user to submit both a password and another form of verification, such as a number delivered to their phone, two-factor authentication adds an extra layer of protection. This ensures that the individual seeking to access the information is who they say they are. By requiring a unique code for each login attempt, one-time passwords offer an extra layer of protection, making it more difficult for a malicious actor to get access.

**Man-in-the-Middle**: Encrypted VPNs and multi-factor authentication are powerful anti-man-in-the-middle defenses. VPNs assist to prevent hostile actors from intercepting sensitive information being transferred by encrypting internet interactions. Endpoint device security, such as that of computers and smartphones, aids in the prevention of malware assaults that may be utilized by man-in-the-middle attackers to obtain unwanted access. By requiring extra kinds of verification, such as a code delivered to a phone, multi-factor authentication, which needs more than just a login and password for access, helps to further guard against man-in-the-middle attacks. Overall, these remedies offer a complete strategy to preventing man-in-the-middle attacks and protecting sensitive data.

The countermeasures used to prevent identity fraud/theft and man-in-the-middle attacks in the application appear to be adequate. Two-factor authentication and the use of one-time passwords delivered through SMS provide an additional layer of security to verify the identity of the user and protect against identity fraud/theft. Encrypted VPNs and multi-factor authentication are effective against man-in-the-middle attacks by encrypting internet interactions and adding extra verification methods. Endpoint device security further strengthens the defense against man-in-the-middle attacks. These countermeasures provide a comprehensive strategy to prevent these types of attacks and protect sensitive data. In conclusion, it the countermeasures used in the application are enough to provide protection against identity fraud/theft and man-in-the-middle attacks. The measures described, such as two-factor authentication, one-time passwords, encrypted VPNs, multi-factor authentication, and endpoint device security, are described as being effective and offering a comprehensive strategy for protection

***8. Listing three of the best practices of coding a secure code***

1- Handle exception.2- Programming language.3- Validate the request.

***9. What is the effect of using the listed best practices (with code)***

**Exception handling:** Exception handling is an important aspect of software development. When an exception is raised, it indicates that something unexpected has occurred during the execution of the program. Without proper exception handling, the program may terminate abruptly, leading to loss of data and a poor user experience. Exception handling involves catching exceptions and dealing with them in a meaningful way. This could involve logging the error, displaying a user-friendly error message, or taking some alternative action that allows the program to continue running. **Programming language**: is crucial for writing efficient and effective code. The more familiar you are with the syntax, libraries, and tools of a language, the more efficiently you can solve problems and write quality code. It also allows for better understanding of code written by others, making maintenance and collaboration easier.

**Request validation**: Request validation is the process of ensuring that the data being sent to a program or system is correct and in the correct format. Invalid or malicious inputs can cause problems such as security vulnerabilities, crashes, or incorrect processing. Validation checks can be performed on incoming data, such as checking the format of user inputs, checking the length of strings, or verifying that numbers are within a certain range. This process helps to ensure data integrity and security by filtering out invalid or malicious inputs.

***10. Assess the implemented best practices to mitigate the security vulnerabilities.***

Exception handling is an important aspect in software development that helps to ensure a stable and smooth user experience. Proper exception handling helps to prevent abrupt termination of the program and the potential loss of data, by catching and dealing with exceptions in a meaningful way.Programming language knowledge is crucial for efficient and effective coding, as a deeper understanding of the syntax, libraries, and tools of a language enables the solving of problems and writing of quality code more efficiently. Additionally, familiarization with a language makes maintenance and collaboration easier.Request validation plays a crucial role in ensuring data integrity and security. By performing validation checks on incoming data, such as checking the format of user inputs, the length of strings, and verifying numbers within a specified range, the system is protected against invalid or malicious inputs that may cause security vulnerabilities, crashes, or incorrect processing.In conclusion, the above points are effective measures in mitigating security vulnerabilities in software development. Exception handling helps to prevent abrupt termination of the program, programming language knowledge enables efficient and effective coding, and request validation ensures data integrity and security by filtering out invalid or malicious inputs.

***11.What dynamic testing techniques did you use? Justify the using of these techniques***.

Dynamic testing is an important part of the software development process. It entails actively running the code, program, or software in order to evaluate its behavior and performance. The primary goal of dynamic testing is to ensure that the implementation fulfills the requirements, specifications, and functions specified during the design process.

This sort of testing takes a hands-on approach to evaluating code and exposing any flaws or defects that static testing approaches, such as code review and documentation analysis, may have missed. Dynamic testing mimics real-world scenarios, allowing engineers to assess the behavior of software under various settings and ensure compliance with quality requirements.

Dynamic testing is carried out throughout the development process, beginning with unit tests to evaluate individual components and ending with integration testing to ensure the components operate fluidly together. The outcomes of dynamic testing are utilized to enhance and repair the code, resulting in a high-quality, dependable, and efficient software product.

**Unit testing:** ensures that individual components of a software system perform as intended in isolation. This approach is useful because it enables developers to detect and solve defects early in the development process, before they become more difficult and costly to fix.

**Integration testing**: ensures that the various components of a software system operate together as planned. This method is required because individual components may work properly on their own, but when integrated with other components, new issues may emerge. Before the product is distributed to consumers, integration testing helps detect and fix these issues.

**System testing:** ensures that a full software system satisfies all of its criteria. This approach is critical for assessing the overall quality and functioning of the system, as well as finding any faults that may have gone undetected during unit and integration testing. To guarantee that the software is evaluated objectively, system testing is often undertaken by a separate testing team.

***12. Evaluate the applied testing techniques if they are enough to say that the application is secure***

An evaluation of the above testing techniques:Unit testing is a crucial method in software development that aims to ensure the individual components of a system are functioning as intended. By catching defects early in the development process, unit testing helps to minimize the cost and complexity of resolving issues.Integration testing is another important aspect of software development that confirms the different components of a system work together effectively. This approach is vital for identifying and fixing any issues that may arise when the components are combined.System testing provides a comprehensive evaluation of the complete software system, allowing for assessment of its overall quality and functionality, as well as detection of any faults that may have been missed during unit and integration testing. To guarantee impartiality, system testing is often performed by a separate testing team.To summarize, the testing techniques mentioned above are crucial in guaranteeing the quality and security of software systems. Unit testing helps detect and resolve issues early in the development phase, integration testing verifies the proper functioning of integrated components, and system testing provides a thorough evaluation of the complete system. Nevertheless, it is essential to understand that testing is not the only method to ensure the security of an application. Other measures such as code review, threat modeling, and penetration testing may also be necessary to ensure the overall security of the application.