Project deliverable 1

1. Introduction

I would like to develop a web project that will host a software vulnerabilities database. If a certain vulnerability has functional exploits developed, the vulnerability page will also contain those. Moreover, people can browse a vulnerability and find which products are affected by it, or browse products and see which vulnerabilities affect a product and if so what versions. I also plan to implement a forum (each user can comment) on each vulnerability or software page. There will be two roles for user: NORMAL\_USER and MODERATOR.

2. Tech stack

For the project I would like to use Spring Boot for the back end, MySQL Server for the database server and React Native for the front end. I choose these technologies because I have worked with them before.

3. Software Architecture

For this project I would like to use the **multitier architecture** implemented using three layers:

* Controller
* Service
* Repository

The **Controller layer** is responsible for exposing the functionality so that it can be accessed by external clients.

The **Service layer** is where all the business logic should go. Here usually all the mathematic operations are done.

The **Repository layer** is responsible for storing and retrieving data from the database.

4. Functional requirements

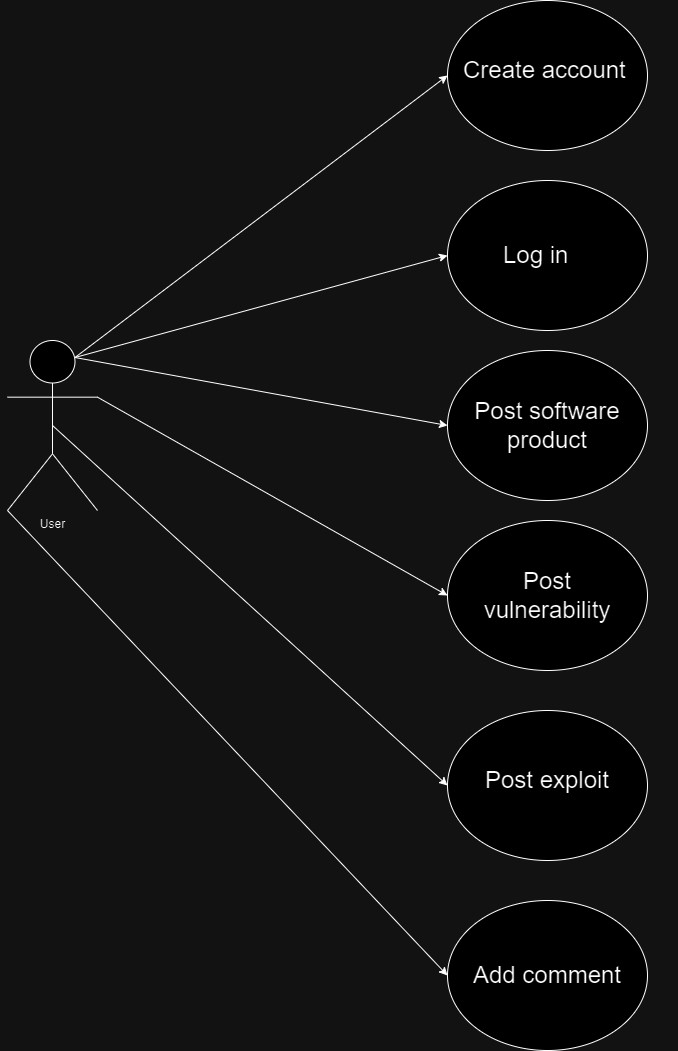
* A person can browse pages of vulnerabilities or software products, or even view the comments even if not logged in
* A user can create an account
* A user can log in
* If logged in a user can do all the following actions
* A user can submit a software product for validation
* A user can submit a vulnerability for a software product for validation
* A user can submit an exploit to a vulnerability for a software product for validation
* A moderator can approve any of the above requests for posting.
* A user can write a comment about a vulnerability.

5. Non-Functional requirements

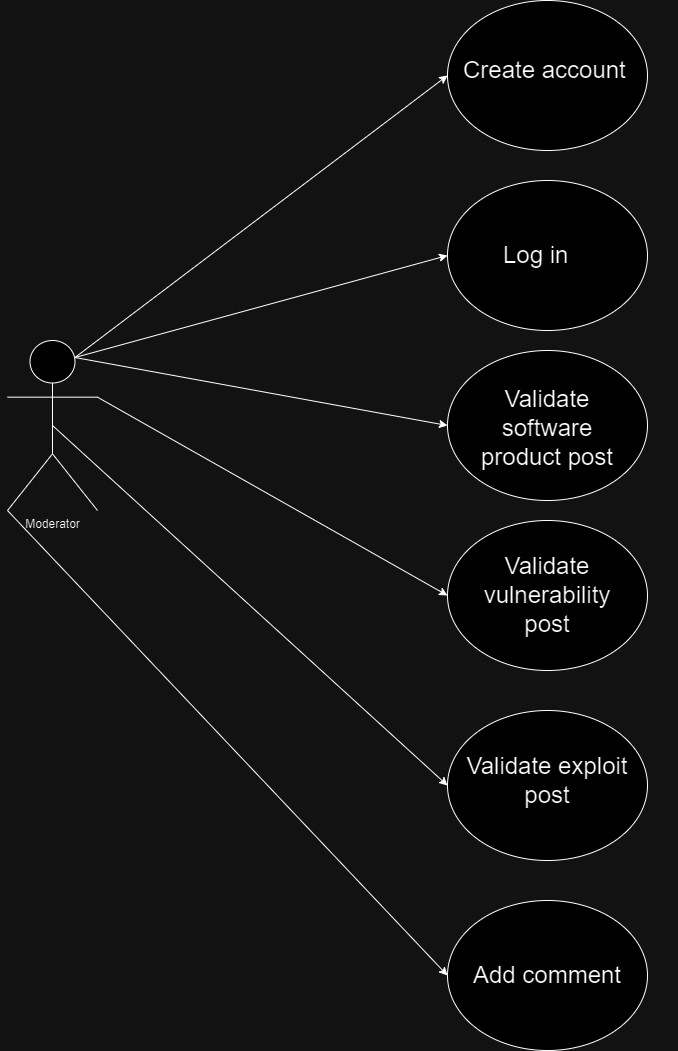
* Speed, the application should execute what it is expected from the user in a reasonable amount of time
* Security, all passwords should be stored encrypted in the database and all requests that alter data should be done only with a valid JWT authentication token
* Capacity, we should ensure that there is enough capacity for a large number of users, vulnerabilities and software programs.
* Usability, we should ensure an effective and easy to use user interface for any type of user.

6. Diagrams

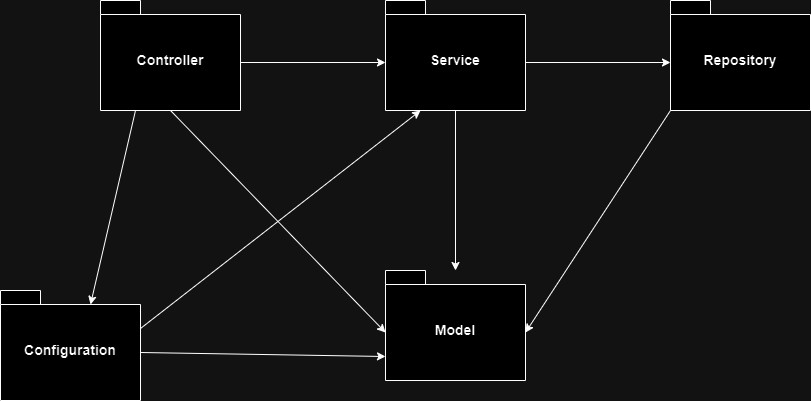
User

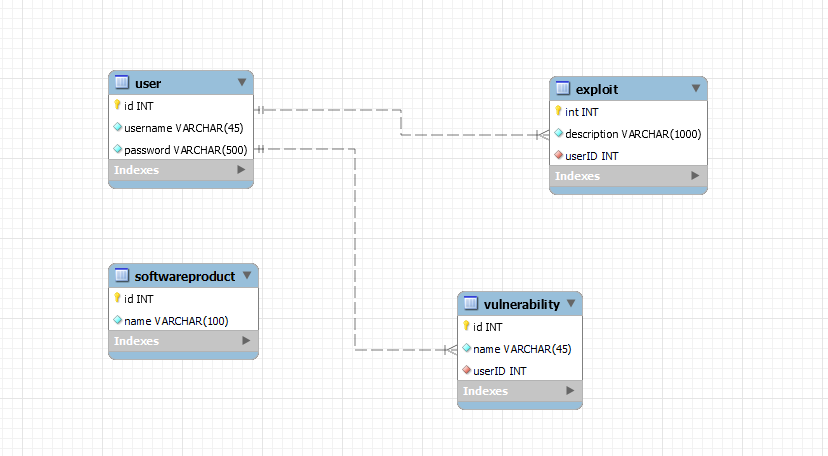


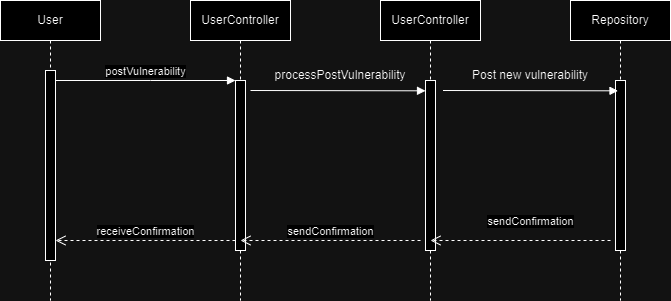
Admin

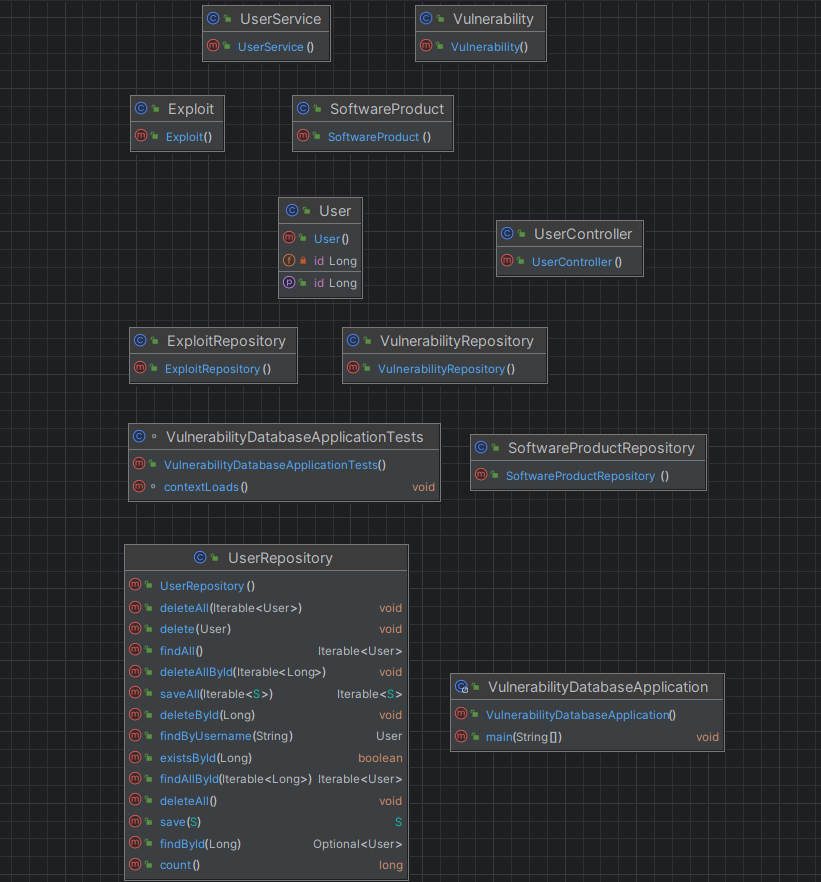


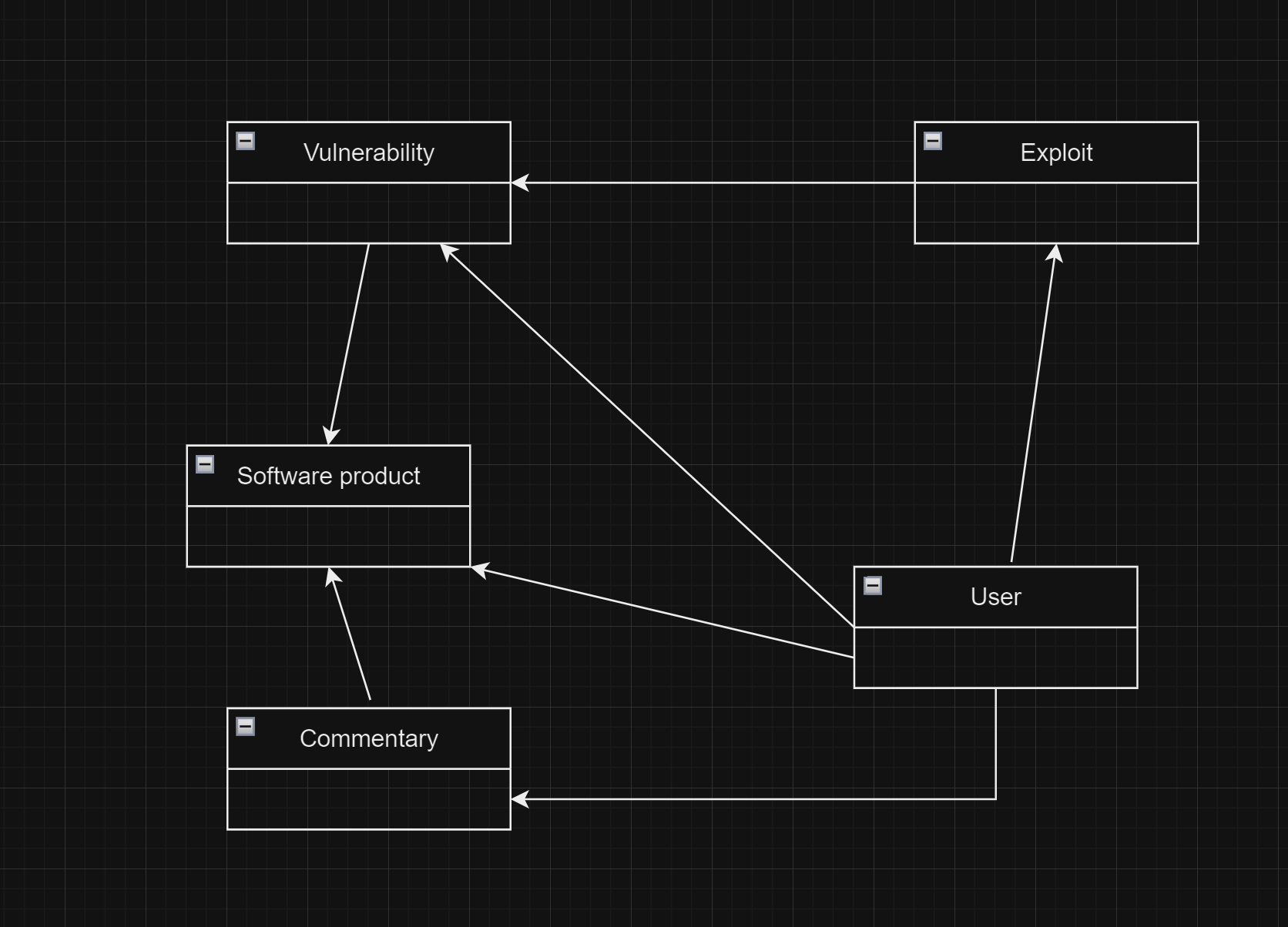
Package Diagram



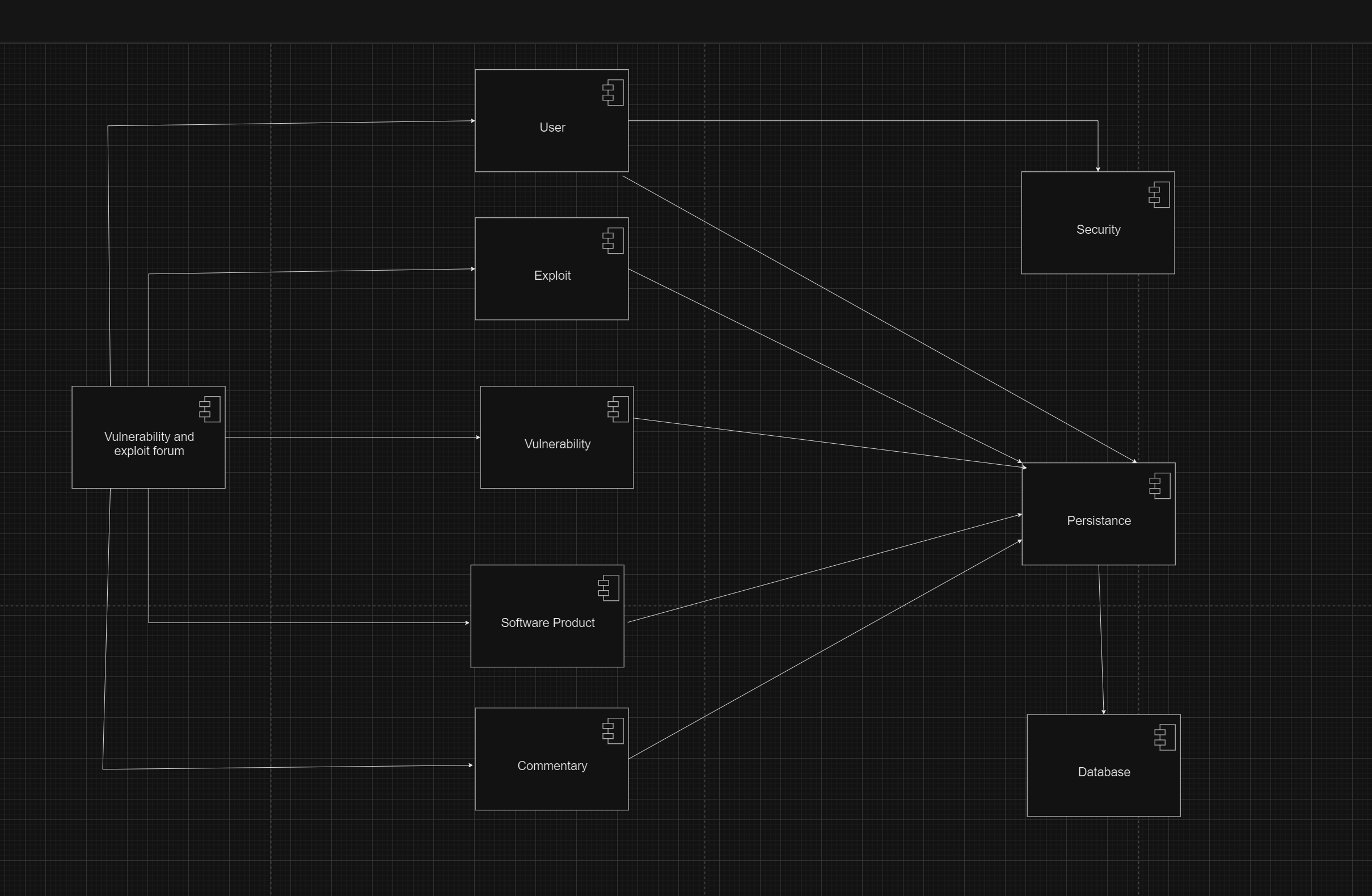






Domain Model

Component Diagram



Deployment Diagram

