**Case Study 9**

Task 1 ) Security concern for product team

* Why Should Product Teams Care About Security?
* Product teams need to incorporate security concerns into their product plans, not as an ancillary service, but as a core feature of their product.
* Why? For one, digital insecurity has become normalized.
* And for another, consumers, businesses, and government agencies are becoming increasingly concerned about the safety and privacy of networks, transactions, and data.
* Three Security Considerations
* Each new platform is a new attack vector.
* Over time, attacks get more sophisticated.
* Frequently, human behaviour is the weakest link in the security chain.

Task 2 ) Application architecture pattern

* Architectural patterns are ways of capturing proven good design structures, so that they can be reused. Software architects have been looking for ways to capture and reuse the architectural knowledge that have proven successful in the past.
* More specifically, an architectural pattern is a package of design decisions that is found repeatedly in practice, has well defined properties that can be reused and describes a class of architectures.
* Developing an architecture can be seen as a process of selecting, tailoring, and combining patterns. The software architect must decide how to instantiate a pattern, how to make it fit with the specific context and the constraints of the problem. And later on we will discuss it more in details.
* Model-view-controller pattern should be use to develop new system.
* Model-view-controller pattern

This pattern, also known as MVC pattern, divides an interactive application in to 3 parts as,

1. **model** — contains the core functionality and data
2. **view** — displays the information to the user (more than one view may be defined)
3. **controller** — handles the input from the user

**Usage**

1. Architecture for World Wide Web applications in major programming languages.
2. Web frameworks such as [**Django**](https://en.wikipedia.org/wiki/Django_(web_framework))and [**Rails**](https://en.wikipedia.org/wiki/Ruby_on_Rails).

Task 3 ) Enterprise application architecture pattern

* The use of architecture pattern increases flexibility, maintainability, and scalability. In a Layered architecture (3 tier ) we separate the user interface from the business logic, and the business logic from the data access logic. Separation of concerns among these logical layers and components is easily achieved with the help of layered architecture.

Task 4 ) **Advantages of layer architecture**

* Layered architecture enables teams to work on different parts of the application parallelly with minimal dependencies on other teams.
* Layered architecture enables develop loosely coupled systems.
* Different components of the application can be independently deployed, maintained, and updated, on different time schedules.
* Layered architecture also makes it possible to configure different levels of security to different components deployed on different boxes. sO Layered architecture, enables you to secure portions of the application behind the firewall and make other components accessible from the Internet.
* Layered architecture also helps you to test the components independently of each other.

**Disadvantages of layer architecture**

* There might be a negative impact on the performance as we have the extra overhead of passing through layers instead of calling a component directly.
* Development of user-intensive applications can sometime take longer if the layering prevents the use of user interface components that directly interact with the database.
* The use of layers helps to control and encapsulate the complexity of large applications, but adds complexity to simple applications.
* Changes to lower level interfaces tend to percolate to higher levels, especially if the relaxed layered approach is used.