Three-tiered software architecture pattern

**Presentation Tier** – Front end layer, Consists of the user interface. Often a graphical interface accessible through a web browser or web-based app. Mainly HTML5, JavaScript, CSS. Displays information related to services available on the website.

**Application Tier** – Contains the functioning business logic, drives an applications core capability. Often Java, .Net, C#, Python, C++.

**Data Tier** – Contains the database/data storage system and data access layer. Often MySQL, Oracle. PostgreSQL, SQL Server.

* Gives flexibility to develop a specific part of the application. We have the ability to replace or upgrade independent tiers.
* The team contains developers who specialize in front-end, back-end and data development. Modularizing the tiers allows for each team to work with minimal impact on the other disciplines.
* Including scalability as a user story means that we need an architecture that is able to accommodate this. The three-tiered pattern allows for each scale to be developed independently depending on the given need.
* Included are multiple testing phases, these testing phases will also test the reliability of the servers and website up time. The use of a three-tiered pattern allows for increase reliability by spreading out the hosting and not relying on a single server. The increased independence that is designed into the pattern minimizes the impact when a server goes down.
* Provides an easy way to maintain the code base, managing presentation and business logic separately.
* Ensures that the product is able to stay up to date and utilize new technologies. There is the option to redesign the application to be ready for the future.