# 3. Development Methodology

## 3.1) Description of the methodology chosen

For the development of this project, I will be using waterfall methodology. The waterfall model is a linear, sequential approach to the software development lifecycle that is popular in product development and software engineering. (Rouse, 2019)

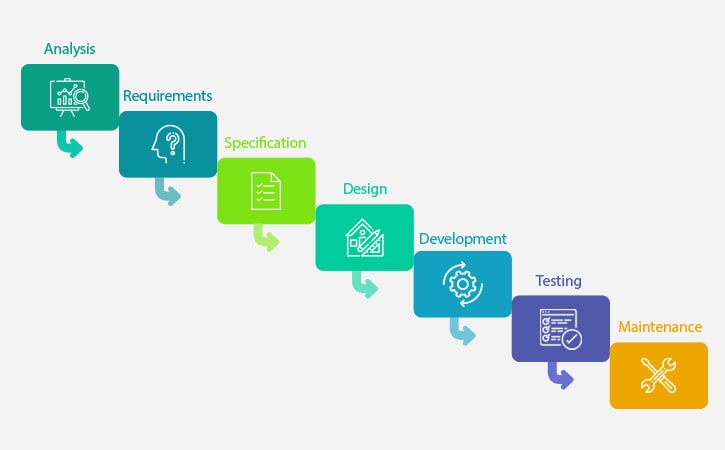


Fig: Waterfall Model

I will choose the waterfall model because of the following reasons:

• It is simple and very easy to understand the processes,

• Milestones, stages, and task are easily defined and arranged,

• Phases are carefully processed and completed one stage at a time,

• Each phase have its own delivery and a reviewing process,

• It is predictive, sequential and planning oriented.

As my project is going to be not very large, I think that waterfall methodology is going to be a good approach. All the requirement is visible and also the user’s participation is not so important for the development of this project. Also, waterfall methodology will help me to make my application good as each phase is started after finishing the current phase.

## 3.2) Design Pattern

For the development of this project, I will be using the Model View Controller (MVC) design pattern.

**Model:** It contains only the pure application of the data. It does not contain any logical describing how to display the information to a user. (Kumar, 2017)

**View:** It displays the model’s data to a user. It knows how to access the model’s data but it does not know what the users can do to alter the data. (Kumar, 2017)

**Controller:** It exists between the model and the view. It controls the data flow into the model and updates when the data is changed. (Tutorials Point, 2017)



Fig: MVC Design Pattern

## 3.3) Architecture

For the development of this project, I will be using Client-Server as an Architecture.

**Client-Server Architecture:**

It is an architecture of a computer network, in which the client requests a service from a centralized server and the centralized server gives service to a remote processor or a client-server.

Advantages:

* Access, resource and the integrity of the data are controlled by the centralized server,
* We can increase the number of client and server separately in the network,
* We can easily replace, repair and upgrade a server without affecting the users.

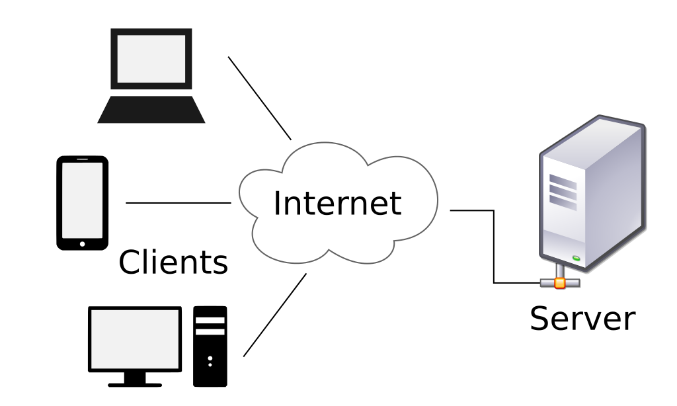


Fig: Client-Server Architecture

In this project, using this architecture client can easily access the resources, data, and files which are stored in the server. It will help to interact front-end (**client**) with users and back-end (**server**) with the shared resources. While deployment of the project any problem may arise, during that time this architecture will act as scalability. Which means it can be scaled in mainly two types they are Horizontal Scaling and Vertical Scaling.

**Horizontal Scaling:** It simply means adding or removing the client’s workstation on a slight performance impact.

**Vertical Scaling:** It simply means upgrading or migrating server to a larger and faster server.

# References

Kumar, S., 2017. *mvc-design-pattern.* [Online]   
Available at: https://www.geeksforgeeks.org/mvc-design-pattern/  
[Accessed 06 April 2019].

Rouse, M., 2019. *waterfall-model.* [Online]   
Available at: https://searchsoftwarequality.techtarget.com/definition/waterfall-model  
[Accessed 06 April 2019].

Tutorials Point, 2017. *design\_pattern/mvc\_pattern.* [Online]   
Available at: https://www.tutorialspoint.com/design\_pattern/mvc\_pattern.htm  
[Accessed 06 April 2019].