**What is the Seperation of Concerns principal?**

**Can you build a View without having a Model?**

No. If you split it up horizonally then you won't have 3 people working on code.

Everyone is safe working in their buckets. If they change code it won't mess up someone elses code.

Controller: Business Logic

If we split it up vertically, we can deliver functionality faster, don’t have to wait for all 3 to show code.

If we split it up horizontally, we must wait for all 3 people to finish.

Separation of Concerns (a reason for change) we want to limit the number of models.

If someone has a new feature you want to be able to change a few things vs 20 different things.

EX: Currently tracking phone numbers. You need to add a new number. What files will you have to change?

The Controllers need to change for an additional piece of data.

**Why is it important?**

**What responsibilities do Views have in MVC websites?**

Presentation logic. To turn the model into something the user can see. What are we showing the users? We are showing data to users. It’s the front end closest to the user. Don’t confuse views with files.

**What responsibilities do Controllers have in MVC websites?**

Passing data from the View to the Model. Handles most of the business logic. Also involved in the routing process.

Controller gets data first. It receives request inputs go to Controller first. First thing Controller does when receives data? First line

in Method is check if Model state is valid. Taking in data validating it and processing it.

They manipulate data. Process request. **Business logic** part of application.

The controller then completes any operation that is needed by using the data that the model is holding then uses those results to display the required view from the operations.

UI Control, button, form, window are controls. Separate from Controllers.

**What responsibilities do Models have MVC websites?**

It’s all about the data. What sets that layer apart from the other 2(View, Controller)?

Responsible for reading the file and writing that file. Responsible for knowing what format data is saved.

views don't save data. Controllers don't do that either validation happens here. They can initiate data, but they don't save data.

EX: Emails are generally considered unique. If Controller doesn't check and the Model just saves it. They will get error message.

Presentation logic Business logic

Data Access part of our applications. This is where the data is held that the views and controllers will need to use throughout their process. Models which contain or represent the data that users work with.

**What is Test-Driven-Development (TDD)?**

Write test of any sort first.

When we write test, we determine what is important to the user. Test-Driven-Development usually uses Unit Test. But doesn't have to.

The key difference is TDD is that tests are written first (before the code being tested)

Usuall you start with the test so you can think through possible problems and hot to fix them.

**How is it different from how developers would normally write code? What are the pros and cons of practicing TDD?**