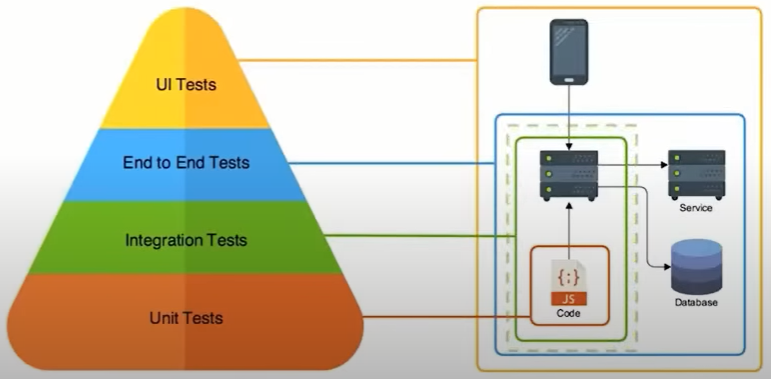
Смотри V model: Unit testing, Integration testing, System testing.

Integration testing is known as the **second level** of the software testing process, following unit testing. Integration testing involves checking individual components or units of a software project to expose(выявить) defects and problems to verify that they work **together** as designed. (уровень тестирования, который фокусируется на взаимодействии между компонентами)

Integration testing is a level of software testing where individual units are combined and the **connectivity or data transfer** between this units is tested.



As a rule, the usual software project consists of numerous **software modules**, many of them built by different [programmers.](https://www.simplilearn.com/how-to-become-programmer-article) Integration testing shows how well these elements work together. After all, each unit may function perfectly on its own, the question is, “Сan they be brought together and work smoothly?”

So, integration testing is the way we find out if the **various parts** of a software application can play well with others!

**Основная идея интеграционного тестирования это протестировать что (готовые) части программы “общаются” между собой так как ожидалось, т.е. узнать правильно ли они интегрированы между собой**

Например написано 2 модуля/блока программы. По окончанию выполнения первого модуля формируется какой то output который является input-ам для второго модуля. Вот здесь интеграционное тестирование и проверяет чтобы всё работало как ожидается и чтобы соединение обеих частей было коррекным.

**Integration testing it is white box testing**

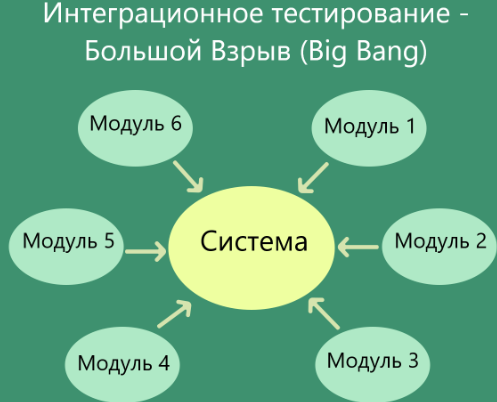
six most popular forms of **integration testing**:

**Big Bang approach**

This approach involves integrating all the modules and components and testing them at once simultaneously as a single unit. The disadvantage is that it is hard to find the source of the problem. Не надо делать какоето модульное тестирование, ждёшь пока всё готово и тестишь всё целиком, с одной стороны экономиш время и тестируешь готовый продукт, но найти причину бага сложно.

или

Все или практически все разработанные модули собираются вместе в виде законченной системы или ее основной части, и затем проводится интеграционное тестирование. Такой подход очень хорош для сохранения времени.



**Hybrid Testing Method**

This method is also called "sandwich testing." It involves simultaneously testing top-level modules with lower-level modules and integrating lower-level modules with top-level modules, and testing them as a system. So, this process is, in essence, a fusion of the bottom-up and top-down testing types.

**Incremental Approach**

This approach integrates two or more logically related modules, then tests them. After this, other related modules are gradually introduced and integrated until all the logically related modules are successfully tested. The tester can use either the top-down or bottom-up methods.

или

при таком подходе тестирование выполняется путем объединения двух или более логически связанных модулей. Затем другие связанные модули поэтапно добавляются и тестируются для правильного функционирования. Процесс продолжается до тех пор, пока все модули не будут соединены и успешно протестированы.

**a) Bottom-Up Approach**

This method requires testing the lower-level modules first (lets called them child), which are then used to facilitate (способствовать) the higher module testing (parent). The process continues until every top-level module is tested. Once all the lower-level modules are successfully tested and integrated, the next level of modules is formed.

или

каждый модуль на более низких уровнях последовательно тестируется с более высокоуровневыми модулями, пока не будут протестированы все модули.

Тоже самое,что и **Top-Down Approach** только наоборот и вместо заглушак ставятся так называемые драйвера-или подпрограммы которые наделены какойто логикой позволяющая тестировать не только нижний модуль но и связь с верхним

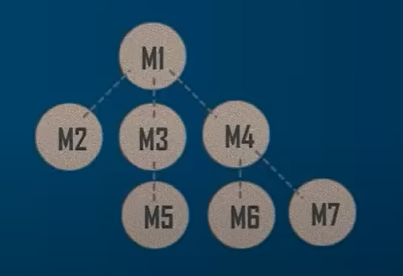


**B)Top-Down Approach**

Unlike the bottom-up method, the top-down approach tests the higher-level modules first, working the way down to the lower-level modules. Testers can use stubs (заглушки) if any lower-level modules aren’t ready

← Здесь модули разрабатываются постепенно наченая с M1 соответственно в тестирование поподёт сначала М1, а на место других модулей будут поставлены заглушки. Т.е. в таком варианте применим **Stubs and drivers-**этот концепт когда какая-то часть программы готова и её надо потестить. Но к примеру это часть логин и она логически связана с частью админЛогин которая не готова, в таком случае применяется заглушка (Stub) вместо админЛогин, и наоборот если готова часть адинЛогин, а логин не готова то вместо логин тоже буде заглушка но называется она driver.

Stubbs is a colled program.

Drivers is a calling program

**С)-** [**Гибридный/сэндвич-подход**](https://www.ques10.com/p/38806/describe-bi-directionalsandwitch-integration-testi/) **(Sandwich/Hybrid/Bi-Directional Approach)**: Представляет собой комбинацию восходящего и нисходящего подходов. Здесь целью является средний слой, в то время как драйверы заменяют верхний слой, а заглушки нижний пока компоненты этих слоев не будут разработаны;

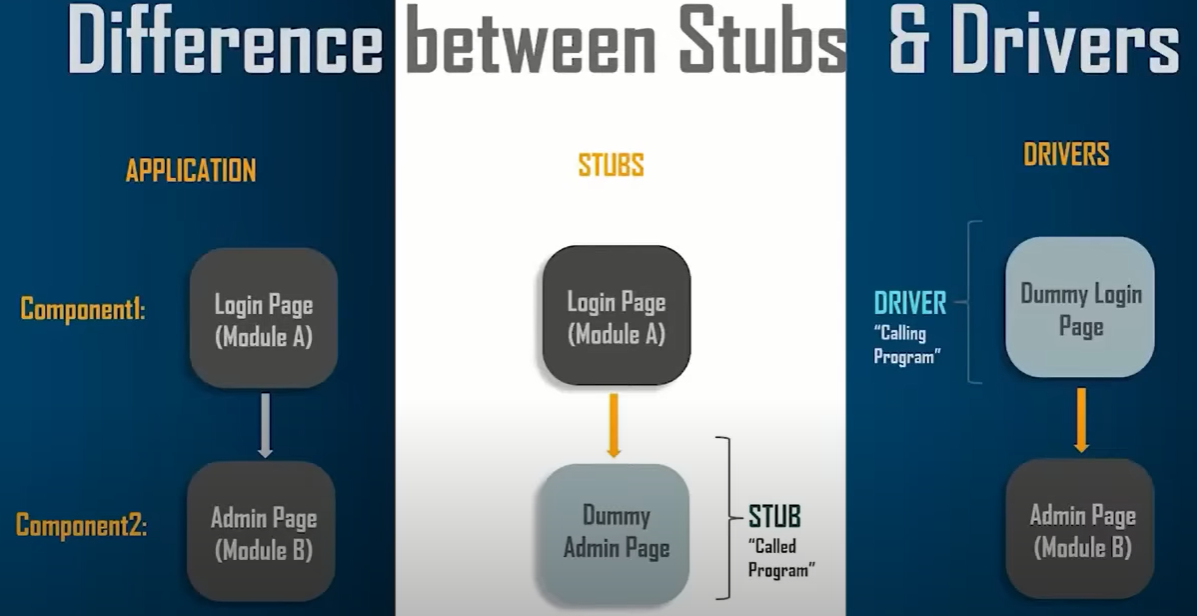
**Non-Incremental approach**

Это когда не по нескольку модулей добавляется, а тестируется всё и сразу. Редко применима так как при возникновении ошибки её сложно найти. Лучше инкрементная модель, добавляю понемногу модули и тестишь.

**Stubs and Drivers**

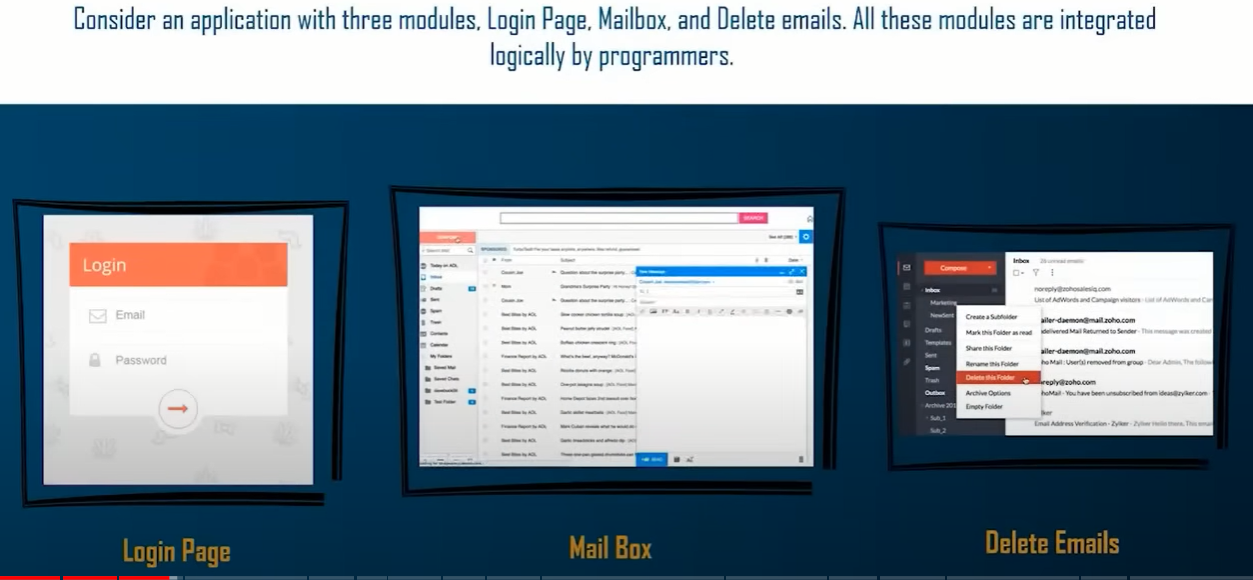
These elements are dummy programs used in integration testing to facilitate software testing activity, acting as substitutes for any missing models in the testing process. These programs don’t implement the missing software module’s entire programming logic, but they do simulate the everyday data communication with the calling module.

Stub - это когда 2 модуля одной программы готова но чтобы они передовали данные между собой, по логике программы должна быть ещё одна часть которая находится должна между двумя уже написанными но бывает так что эта часть 3-яя по какойто причине не готова и чтобы не ждать разроботки 3 части а уже тестировать интеграцию уже написанных частей создаётся что-то вроде временной программы которая не имеет 100% логики но которая обеспечивает возможность протестить имеющиеся 2 части. Такая временная программа называется **stub** (заглушка)

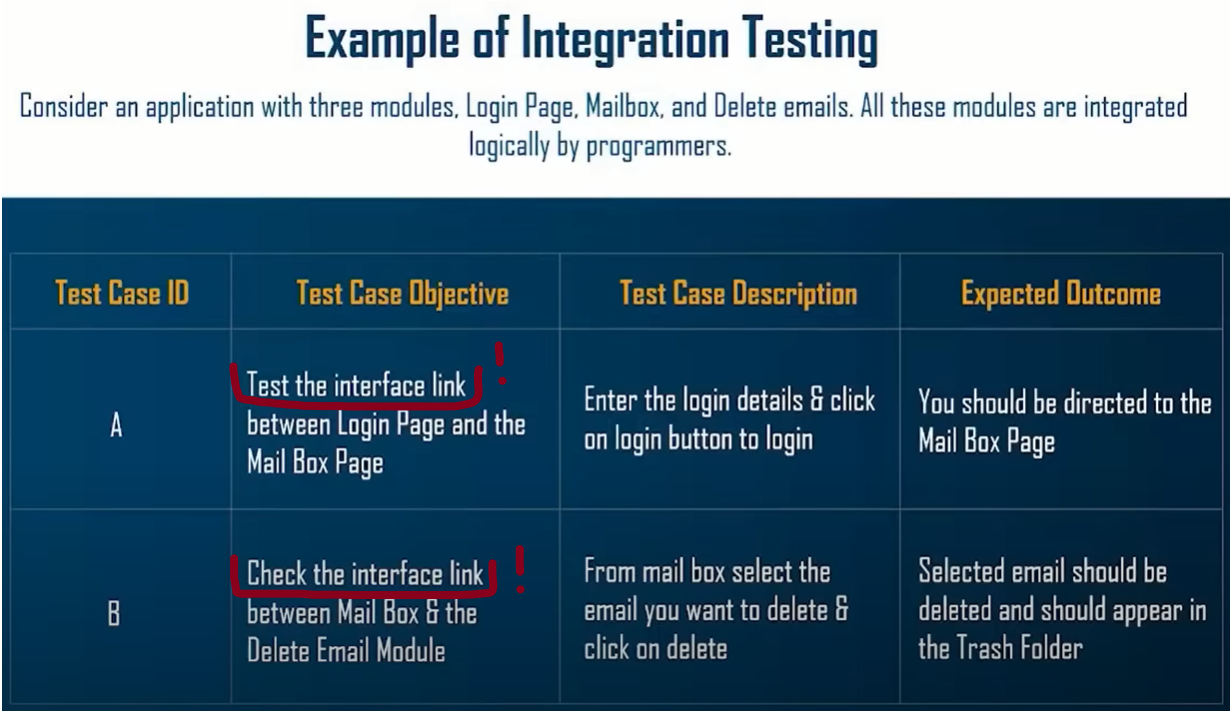


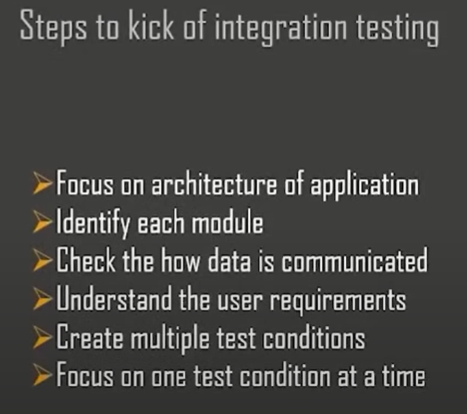
ПРИМЕР:

Ниже скрин 3-х модулей, тестировать по отдельности (unit testing) не надо так как это уже выполнено, теперь нужно сосредоточится на взаимодействии всех частей между собой. Выполнить интеграционное тестирование.



Ниже расположено описание примерных шагов по тестированию данного приложения, как и указывалось раньше основное направление это тестирование СВЯЗЕЙ/ВЗАИМОДЕЙСТВИЯ частей программы





Здесь указаны рекомендации при использовании данного вида тестирования