PL (уровень представления)

PL отображает пользовательский интерфейс и упрощает взаимодействие с пользователем. Уровень представления имеет компоненты пользовательского интерфейса, которые визуализируют и показывают данные для пользователей. Также существуют компоненты пользовательского процесса, которые задают взаимодействие с пользователем. PL предоставляет всю необходимую информацию клиентской стороне. Основная цель уровня представления - получить входные данные, обработать запросы пользователей, отправить их в службу данных и показать результаты.

DSL (уровень обслуживания данных)

DSL передает данные, обработанные уровнем бизнес-логики, на уровень представления. Этот уровень гарантирует безопасность данных, изолируя бизнес-логику со стороны клиента.

BLL (уровень бизнес-логики)

BLL несет ответственность за надлежащий обмен данными. Этот уровень определяет логику бизнес-операций и правил. Вход на сайт - это пример уровня бизнес-логики.

DAL (уровень доступа к данным)

DAL предлагает упрощенный доступ к данным, хранящимся в постоянных хранилищах, таких как двоичные файлы и файлы XML. Уровень доступа к данным также управляет операциями CRUD - создание, чтение, обновление, удаление.

Особенности тестирования монолитных веб-приложений

Монолитная архитектура — это традиционная модель постройки ПО, которая представляет собой единый модуль, который работает независимо от других приложений. Все бизнес задачи приложения, построенного на базе монолитной архитектуры, объединены в единой базе кода. Чтобы внести какие-либо изменения в такое приложение, необходимо вносить изменения во все приложение целиком, что требует много времени.
К преимуществам такой архитектуры можно отнести скорость разработки и простоту тестирования. Обычно проводят сквозное тестирование для проверки ПО от начала до конца E2E (end 2 end). Так как в монолитной архитектуре вся система может пострадать, если откажет хотя бы одна подсистема, таким образом, посредством сквозного тестирования подобных проблем можно избежать.

Процесс сквозного тестирования:

- Изучение требований к сквозному тестированию;
- Настройка тестовой среды и требования к оборудованию и ПО;
- Описание всех процессов системы и ее подсистем;
- Описание ролей и ответственности для всех систем;
- Методология и стандарты тестирования;
- Отслеживание требований и разработка тест-кейсов;
- Входные и выходные данные для каждой системы.

Особенности тестирования микро-серверных веб-приложений

Так как микросервисная архитектура подразумевает, что разные части приложения могут быть написаны на разных языках программирования, а также могут находится на разных серверах, то и процесс тестирование таких сервисов усложняется.
Плюсом является тот факт, что в микро-сервисной архитектуре можно обновить отдельный микросервис и протестировать его не затронув другие сервисы.
Типы тестирования:

- *Unit-тестирование*
- *Контрактное*
- *Интеграционное*
- *e2e*
- *Нагрузочное*
- *Функциональное*