Обычно дефект включает в себя следующие пункты:

1. **Название** — должно отвечать на вопросы “Что? Где? Когда?”, то есть суть проявления дефекта на прикладном уровне, чтобы проблема была понятно не только разработчику и тестировщику, но и отделу продаж или директору. Ведь они не обязаны быть знакомы со всеми техническими деталями продукта. По названию дефекта должно быть понятно к какому функционалу он относится;
2. **Приоритет (Priority)** — степень важности, присваиваемая дефекту. Чем выше приоритет дефекта, тем быстрее приступят к его исправлению.
3. **Критичность (severity)** — на сколько важно воздействие какого-то конкретного дефекта на функционирование компонента или систему. Чем критичнее дефект, тем выше он и важнее с точки зрения влияния на работоспособность ПО.
4. **Описание:**
   Информация о стенде — машина, на которой был замечен дефект;

   Шаги воспроизведения — прежде чем направлять дефект разработчику для исправления, его следует воспроизвести и выделить все условия, которые необходимо соблюсти для его воспроизведения. Если есть какие-то факторы, которые не влияют на воспроизведение дефекта, то об этом следует упомянуть в описании, так как это может сэкономить время разработчика при исправлении. Описание дефекта должно содержать достаточное для воспроизведения количество информации. Также необходимо помнить, что не все дефекты исправляются сразу, поэтому описывать их нужно таким образом, чтобы даже спустя много времени было понятно о чем идет речь, и можно было быстро разобраться с проблемой при ее возникновении. Не стоит забывать также, что команда, которая работает над продуктом тоже претерпевает изменения, то есть новым участникам проекта описание тоже должно быть понятно. Без адекватного описания на актуализацию дефекта может уходить уйма времени.
   Также следует учесть, что программисты, которым предстоит исправлять дефект, могут не быть знакомы с используемым функционалом. Это происходит в командах, где программистов больше двух.
   При частичном исправлении ошибки и повторном заведении

дефекта, стоит оставить информацию о частичных исправлениях (например, использовать перечеркнутый шрифт для того, что было исправлено).
Дефекты следует заводить как можно быстрее. Время играет очень важную роль, так как чем быстрее они будут заведены, тем быстрее будут исправлены.
Мелкие дефекты тоже не следует пропускать. Если существует внутренняя договоренность о том, что мелкие дефекты не заводятся в багтрекер, то можно вести их другим способом (Гугл докс, эксел и т.д.).

Фактический и ожидаемый результат — первый описывает поведение системы на момент обнаружения дефекта, чаще всего содержит описание некорректного поведения, второй — описывает как именно должна работать система в соответствии с документацией.

Жизненный цикл отчета о дефекте и переходы между состояниями

Закрыт (Closed) — означает, что по дефекту не планируется больше никаких действий;
“Не является дефектом” — приложение так и должно работать и описанное поведение не является аномальным;
“Дубликат” — дефект уже описан в другом отчете;
“Не удалось воспроизвести” — разработчику не удалось воспроизвести дефект на своем оборудовании;
“Невозможно исправить” — существует причина, по которой команда разработчиков не может исправить дефект, так как это вне полномочий команды;
Открыт заново (ripened) — в данное состояние дефект может перевести тестировщик, который убедился в том, что дефект не был исправлен или был исправлен не до конца;
Рекомендован к отклонению (to be declined) — в это состояние отчет о дефекте может быть переведен из множества других состояний с целью вынести на рассмотрение вопрос об отклонении отчета;
Отклонен (declined) — используется тогда же, когда и состояние “Закрыт”;
Отложен (deferred) — если исправление дефекта в данный момент представляется нецелесообразным или не представляется возможным, но, возможно, в скором времени ситуация поменяется (выйдет новая

версия библиотеки, выйдет из отпуска специалист по данной технологии, изменятся требования заказчика и т.д.).