Git Flow & Github Flow

Git Flow

Git Flow는 정식 코스 요리를 내는 대형 레스토랑에 비유할 수 있다. 이는 여러 단계의 검증을 거쳐 정해진 날짜에 맞춰 소프트웨어의 새 버전을 출시하는 방식과 유사하다.

master 브랜치는 손님에게 제공되는 완벽히 검증된 최종 메뉴판에 해당한다. 함부로 수정하지 않는다.

develop 브랜치는 신메뉴 개발이 이루어지는 메인 주방이다. 다음 버전에 포함될 기능들이 이곳에서 종합적으로 준비된다.

feature 브랜치는 요리사 개인이 새로운 메뉴를 개발하는 작업 공간이다. 개발이 완료되면 메인 주방인 develop 브랜치로 합쳐진다.

release 브랜치는 신메뉴 출시 직전, 최종적으로 맛을 보고 점검하는 단계이다. 이 과정에서 발견된 사소한 문제들은 여기서 바로 수정한다.

hotfix 브랜치는 이미 판매 중인 메뉴에 문제가 발생했을 때, 다른 작업에 영향을 주지 않고 해당 문제만 긴급하게 수정하는 역할을 한다.

이러한 흐름은 새로운 기능(feature)을 개발하여 주방(develop)에 통합하고, 이것들이 모이면 정식 출시(master) 전에 최종 점검(release)을 거치는 체계적인 방식이다. 과정이 명확하지만 다소 복잡하고 속도가 느릴 수 있다.

Github Flow

master 브랜치는 푸드트럭의 오늘의 메뉴판과 같다. 이 메뉴판은 지금 바로 주문 가능한 메뉴를 의미하며, 항상 최신 상태를 유지한다.

feature 브랜치는 스페셜 메뉴를 준비하는 과정에 해당한다. 새로운 기능의 개발이 여기서 이루어진다.

흐름은 매우 단순하다. 새로운 메뉴(feature)를 만들고, 동료에게 확인받은(Pull Request) 후, 승인되면 바로 오늘의 메뉴판(master)에 추가하여 판매를 시작한다. '다음 시즌 메뉴 준비'나 '출시 전 최종 점검' 같은 복잡한 단계가 없다.

Difference

두 방식의 가장 큰 차이점은 다음과 같다.

첫째, 목표가 다르다. Git Flow는 정해진 주기에 맞춰 완성도 높은 '버전'을 출시하는 것을 목표로 한다. GitHub Flow는 좋은 기능이 생길 때마다 즉시 반영하여 서비스를 계속 최신 상태로 유지하는 것이 목표다.

둘째, 브랜치의 복잡성이다. Git Flow는 역할이 명확히 나뉜 여러 브랜치를 사용하여 복잡하지만 안정적이다. GitHub Flow는 master와 feature 단 두 종류의 브랜치만 사용하여 매우 단순하고 빠르다.

셋째, 안정성 확보 방식이 다르다. Git Flow는 여러 단계의 브랜치를 거치며 안정성을 확보한다. GitHub Flow는 master 브랜치에 합치기 전, 동료의 꼼꼼한 코드 리뷰와 자동화된 테스트를 통해 안정성을 확보한다.

Conclusion

결론적으로, Git Flow는 출시 일정이 정해져 있고 버전 관리가 중요한 대규모 프로젝트에 적합하다. GitHub Flow는 수시로 업데이트가 필요한 웹 서비스나, 빠르고 민첩하게 개발해야 하는 소규모 프로젝트에 더 적합하다.