### **Merge vào master**

Yay! Sau một thời gian quằn quại, cuối cùng team của bạn đã ra được sản phẩm tương đối ổn. Giờ là lúc merge vào master và triển khai lên server.

Lúc này, cũng như khi merge vào dev, bạn có thể chọn merge (fast-forward hoặc non-fast-forward) hay rebase, squash và merge, nhưng theo kinh nghiệm của Ehkoo, merge fast-forward sẽ là lựa chọn tốt nhất, giúp cho master và dev luôn song song với nhau.

### **Hotfix**

Hôm nay, thứ 6, ngày 1X. Bạn chạy npm run build rồi rsync code ở master lên server. Mọi thứ hoàn toàn bình thường. Bạn vào website, click vài cái. "Ngon, chạy rồi", bạn thầm nghĩ, “đi nhậu thoy!” Nhưng vừa vươn vai định gọi điện cho gấu, thì "ó e ò e", chuông điện thoại reng, số máy của sếp. “Hí hí, chắc được thưởng nóng chăng?” Bạn bắt máy, và nghe giọng sếp âu yếm GẦM ở đầu dây: “LỖI RỒI MÁÁÁ!!!”

Xin đừng trụy tim. Hãy hít một hơi thật sâu, rồi bình tĩnh tạo một branch mới từ master, fix-xxxchẳng hạn. Nhờ lắng nghe tiếng sếp gầm, bạn đã mau chóng mò ra lỗi trong đống code (vì bạn là dev cứng mà hihi). Bạn khẽ rủa thầm ku đào nhí viết code không kĩ, tự rủa nhẹ bản thân vì review sót. Nhưng thôi kệ, fix nhanh rồi còn về, kẻo gấu xé xác T^T.

Bạn bèn merge --no-ff nhánh fix-xxx vào cả hai nhánh master và dev. Bằng cách này, phần sửa lỗi sẽ xuất hiện ở cả hai branches, giúp history không bị rẽ nhánh bất ngờ.

Bạn push, và chuông điện thoại lại vang lên…

### **Vài vấn đề linh tinh khác**

#### Có cần nhánh staging không?

Trong một số dự án, ngoài dev, còn có một đội ngũ “thần bí” được gọi là QA/QC. Họ được sinh ra trong team là để bới lỗi của anh em nhà dev, nên quan hệ đôi bên không được tình thương mến thương cho lắm. Dầu vậy, họ vẫn cần một nhánh riêng có tên gọi staging. Nhánh này sẽ chứa phần code ở giữa master và dev. staging được tách ra từ dev, có nhiều tính năng hơn master, và tương đối ổn định để có thể merge vào master.