1. Mi percepción de lo que es la calidad de software es: Un software que cumple no solo con las especificaciones que nos pide el cliente que tenga, sinó que también permite realizar cosas como mantener el sistema fácilmente, o que el mismo sea robusto y confiable, es decir, que funcione cuando lo necesitamos. Un software de mala calidad es aquel que falla cuando hay un evento inesperado, que no anda cuando lo necesitamos (como las impresoras, que nunca andan cuando uno realmente las necesita o cuando está apurado). Un software de calidad por lo tanto responde rápidamente, o por lo menos informa de que va a demorar (un ejemplo sencillo sería un página de carga en un sitio web o la barra del estado de descarga, que dicen “estoy demorando, pero algo estoy haciendo”). Algo de calidad se VE de calidad, un concepto que es más sencillo de percibir en el hardware, pero que en el software se “tradujo” al UX, no muchas personas saben lo que es, pero todas la valoran y lo esperan. La calidad del software entonces, es algo que da valor, seguridad, confianza, que “transpira” buen trabajo y esfuerzo.

2.

3
6
8 Estas pruebas pueden ir más allá de las pruebas unitarias. Estas pueden ser de integración, de carga, de estrés, de integridad de datos, etc.
9
11
12

3. a
   3.1. Intuitiva
   3.2. Eficiencia
   3.3. Riqueza, pero el proveer un conjunto MUY abundante de características, puede resultar **poco intuitivo** (como Facebook hace unos años, que tenía en la pantalla principal demasiadas características y termina confundiendo a los usuarios, sobre todo a los principiantes. Esto fue corregido)
   3.4. Idem a 3.2
   3.5. Robustez
   3.6. Intuitivo
   3.7. Eficiencia
   3.8. Robustez
   3.9. Riqueza y Eficiencia. Es un software personalizable y que por tanto puede ofrecer muchas características (riqueza) y permite que el usuario ajuste a sus necesidades la interfaz, lo que la vuelve muy eficaz (si el usuario la organiza bien)

4. Para mi una empresa que desarrolló su producto con la filosofía de “suficientemente bueno” es: Meta
   Ellos plantearon dentro de Instagram una funcionalidad llamada Reels. En sus inicios los Reels (copia de TikTok) no funcionaban del todo bien y tenían detalles de la UX que hacían que ver videos en su plataforma, en ese formato, no sea de la mejor calidad. Como por ejemplo que la reproduccion de los videos se detenga repentinamente y que la pantalla quede en negro. Sin embargo, el desarrollo era lo

suficientemente bueno como para que los usuarios lo empiecen a usar (ya que cumplia con lo que el usuario queria: ver videos cortos que le generen interés y poder interactuar con ellos y sus creadores), para dar a conocer esa nueva funcionalidad y para, en un futuro, quitarle mercado a TikTok (como hicieron con las historias y SnapChat). Posteriormente el producto mejoró pero sigue siendo lo "suficientemente bueno".

5.

5.1. **F** El trabajo del QA va más allá de correr tests unitarios, ellos son personas que buscan y tratan riesgos de fallo del sistema, teniendo en cuenta tanto reglas escritas (tests unitarios) como estándares, buenas prácticas, integridad de los datos y del sistema, interdependencias, etc. Ellos no se fijan solo en que el código "funcione como debe", sinó que también se fijan que es lo correcto para el objetivo de la empresa. Pueden sugerir correcciones como por ejemplo implementaciones más escalables o seguras, para efectivamente aumentar la calidad del software.

5.2. V

5.3. **F** Las metodologías ágiles no aseguran nada, son solo un marco de trabajo. Dado que la naturaleza iterativa de este tipo de metodologías permite encontrar y corregir fallos en etapas más tempranas de desarrollo, PUEDE pasar (no está asegurado), que el desarrollo termine siendo más rápido. Pero este es, en realidad, un mito sobre las metodologías ágiles.

5.4. **F** Es una buena práctica tener un entorno de **testeo** o **preproducción** que sea lo más parecido posible a **producción**, para poder correr pruebas que nos aseguren con un mayor grado de confianza que las nuevas funcionalidades que estamos desarrollando van a performar correctamente, o como es esperado que lo haga. En estos entornos de testeo se utilizan datos similares a los de producción o se mantiene el entorno en un estado similar. Al fin y al cabo, este entorno nos permite "simular" que estamos probando el código en producción.

6. A raíz de la lectura del artículo "La calidad de los productos Microsoft", se podría concluir que Microsoft necesitaba hacer controles y pruebas más rigurosas de sus productos antes de lanzarlos al público. Posiblemente hayan tenido un *bias* por ser desarrolladores mayormente de habla inglesa, en cuyo caso deberían haber llamado a un usuario de habla hispana o deberían haber asumido ese rol para uno de sus tests. De esa manera podrían haber evitado un error (ahora) tan conocido como ese.
Otro de las fallas que menciona sobre los productos de Microsoft es su falta de seguridad. Esto fue "corregido" cuando introdujeron su propio antivirus, dando una sensación de mayor robustez en el sistema. Sin embargo, no sirve para ejercer un control sobre la conciencia de sus usuarios sobre seguridad.
También menciona las interminables actualizaciones de software. Estas son famosas incluso hoy en día, dando una actualización de seguridad todos los martes, junto con otras fixes mezcladas en el medio.
Un equipo de QA podría haber evitado o mitigado varios de estos errores (los de excel y seguridad) incorporando pruebas de portabilidad (entre idiomas), estándares

de programación segura, probando el software en distintos tipos de entorno o estableciendo estrategias de testeo completas, por ejemplo.