Jakob Nielsen é um cientista da computação que estuda a interação homem-máquina e elaborou um conjunto de heurísticas, ou seja, regras de ouro, para projetar ambientes digitais. Abaixo há uma lista com cada uma delas:

1 - Visibilidade do Status do Sistema: O ser humano é muito dependente da visão em situações do dia a dia. No ambiente digital é mais dependente ainda, já que as outras formas de interação são mais precárias. Por conta disso é primordial que a interface projetada ajude o usuário em relação à sua posição dentro de um sistema digital. Exemplo: quando o usuário acessa uma página específica de um sistema e há nela uma indicação de título assinalando a posição ao usuário. O designer deve sempre deixar com que o usuário tenha sensação de controle e não fique perdido na plataforma.

2 - Compatibilidade entre o sistema e o mundo real: É importante que um sistema digital fale a mesma linguagem do usuário, ou seja, deve-se levar em conta a linguagem que o usuário está acostumado para que consiga se comunicar com a interface de maneira eficiente. Uma maneira de atingir esse objetivo com eficiência é utilizando ícones, que são responsáveis por transmitir uma linguagem universal de fácil reconhecimento.

3 - Controle e liberdade para o usuário: Quando o usuário realiza uma ação por engano deve-se apontar uma direção e uma maneira de solucionar o erro facilmente. Isso dá mais controle para o usuário através de uma "saída de emergência" que leva do estado no qual ele entrou para um estado seguro. Uma maneira de simbolizar essa heurística seria a situação, por exemplo, de um usuário deletar um e-mail importante por engano mas conseguiu encontrá-lo em sua lixeira antes de excluir definitivamente.

4 - Consistência e Padronização: É muito importante manter a consistência das telas ao longo da interface. Dessa maneira não fica necessário que o usuário entenda diversos padrões e formas de interações diferentes para cada tela de um sistema. Com a consistência e padronização é criado um padrão replicado em todo o sistema no qual o usuário apenas precisa memorizar uma vez e isso se repetirá ao longo da plataforma inteira, criando uma melhor experiência de navegação.

5 - Prevenção de erros: O usuário diversas vezes comete erros na navegação de uma interface. A prevenção de erros consiste em tentar impedir que o usuário cometa erros, como por exemplo, alertando que um campo está vazio ou preenchido de forma incorreta antes mesmo de o usuário submeter um formulário. Outro bom exemplo é exibir um alerta sinalizando que a ação que o usuário está tentando executar é destrutiva e solicitar confirmação para realizar a ação.

6 - Reconhecimento ao invés de memorização: É mais simples para o usuário reconhecer padrões ao invés de memorizá-los. Isso acontece porque cérebro é ótimo em reconhecer padrões e, na medida em que objetos são expostos ao usuários, mais as ações ficam familiarizadas e mais fácil ele reconhecerá tal elemento. Para isso é importante ter coesão visual na interface, criando padrões reconhecíveis para os usuários.

7 - Eficiência e flexibilidade de uso: A interface deve ser criada tanto para usuário leigos quanto para experientes. Os leigos precisam de mais tempo e informações para executar tarefas

e realizar ações. Conforme esses usuários vão conhecendo mais a interface, cresce a necessidade de executar as mesmas tarefas por caminhos mais rápidos. É o caso de alguns atalhos em programas, nos quais, por exemplo, há, no menu, a opção de desfazer uma ação e um atalho (Control + Z) para isto.

8 - Estética e design minimalista: Um design minimalista garante que o usuário encontre as informações na plataforma de forma mais rápida e sem se perder. Quanto mais elementos estiverem na interface, maior a chance dele se desconcentrar e deixar de realizar a ação desejada. Assim, o design minimalista torna-se muito eficiente ao transmitir as informações de um sistema.

9 - Ajude os usuários a reconhecerem, diagnosticarem e recuperarem-se de erros: Toda vez que o usuário comete um erro é importante mostrar para ele uma maneira de recuperar. Por exemplo: há uma informação obrigatória em um campo de formulário e o usuário, ao tentar passar por ela, é impedido. Mesmo com o erro, o usuário tem clareza em como se recuperar. Neste exemplo de erro, poderia aparecer uma instrução escrita de recuperação, como " `Esse telefone está sem o DDD, insira-o por favor.` ". O usuário comete um erro e usa a ação Control + Z do teclado. Ou ainda, o usuário exclui um artigo, então o sistema exibe em tela um atalho para desfazer a ação caso seja feita por engano.

10 - Ajuda e documentação: Dentro de um sistema é importante que haja um local com ajuda disponível. Isso faz com que os usuários se sintam seguros caso surja alguma dificuldade.