On the article concerning poor shopping cart design aka "The Oatmeal" we're taught the importance of good web-programming practice for any up and coming web programmer who decides to design a website for an exotic meat company – please don’t design one for grizzly bear meat, that’ll be weird.

Rule # one, don’t forget to mark important fields with asterisk and to bolden the font of the text, otherwise you might end upsetting an overly sensitive CEO who sues the company for poor web-programming - don’t worry, worst-case scenario for first time offenders.

Rule # two, avoid the urge to mandate profile creation and logging in to use the website. The last thing any busy individual needs is another website which screams “join my account group, use me once, and probably never have to use me again”.

Rule # three, avoid the desire to create personalized captchas, and just use re-captcha to better users’ experience and prevent the company from having to receive angry reviews.

Rule # four, align info elements evenly to make the webpage appear more structured, better to place those elements over the input boxes. Neat and structured website equals good reviews which equals increased wage bonuses. Also, use autofill to prevent retyping on already entered information.

Rule # five, be sensitive to common information concerning majority of the website’s users, such as country of origin, to make it easily accessible where needed, and in case of mistyped or invalid entries, don’t remove all the information already provided by users, just those provided incorrectly. You tend to get less abusive users that way.

Rule # six, don’t make users’ review mandatory – keeps from having your feelings hurt by some online troll.

Rule # seven, as previously stated, use auto-form fill so users don’t have to retype everything when they need to edit their order just as they’re about to checkout.

As for the article concerning the three signs of poor web developments, we’re shown three schoolboy error that even professional web-developers make unknowingly. These include: preventing users from using valid email addresses; sending passwords to users as plaintext; limiting length of passwords.

To start, by designing address validators off-the shelf rather than taking advantage of those already in existence, the web-programmer limits the options available to users by preventing them from making use of symbols in typing valid email addresses.

Next, by sending passwords to users as plaintext rather than as an encrypted format, web-programmers expose users’ data to third parties, whether insecure email system, viral programs or unencrypted Wi-Fi. Furthermore, most passwords are reused by users’, and so just getting the password once can lead to compromise of multiple connected systems. Thus, good hashing algorithms, like SHA2, should be used to encrypt passwords securely.

Finally, by limiting the length of passwords, users are unable to securely create complex passwords. With longer passwords providing more security, limiting users’ password length limits how secure web-programmer’s system is. Thus, already existing machines, such as SHA-512, should be used to allow users whatever length of password they desire by storing passwords as hash in database.