

Re: Is there an obvious way to do this in python?

Re: Is there an obvious way to do this in python?

Source: <http://coding.derkeiler.com/Archive/Python/comp.lang.python/2006-08/msg01850.html>

- *From:* "H J van Rooyen" <[mail@xxxxxxxxxxxxxxxxx](mailto:xxxxxxxxxxxxxxxxx)>
 - *Date:* Thu, 3 Aug 2006 14:05:19 +0200
-

"Yu-Xi Lim" <yuxi@xxxxxxxxxxxxxxxxx> wrote:

| H J van Rooyen wrote:

| > |

| > |> And I would like to make this flexible, so that it becomes easy to introduce

| > new

| > |> transactions, without having to run around updating the code in all the user

| > |> machines, with the concomitant version number hassles.

| > |

| > |Then you want a web front end.

| >

| > This seems to me to assume that the server does all the work

|

| Depends on what you mean by "all the work."

What I mean by this is that the server does stuff that I think belongs on the client –

like getting involved in the nitty gritty of what the client should display –

I want the client to be smart enough to do the assembly of the elements of a transaction

by itself, going back to the server for data only when its needed – remember this is essentially an

accounting type data entry package – most of the stuff is typed in as text, when processing documents from the outside, while the locally produced docs like invoices would be generated by the system, to a large extent by making choices amongst alternatives known to the server –

so I see the client interacting with the server quite a lot, eventually to be able do things like auto completion of things like stock codes and descriptions, customer details, etc. – but I don't want every keystroke flying over the LAN and

being handled by the server...

In a sense I want to build an architecture that assembles a record from a mix of user input and user choices out of alternatives (which unfortunately would have

Re: Is there an obvious way to do this in python?

to be kept on the server) and that then ships this to the server to process, and to keep track of – such a record would be either accepted or rejected by the server, and it would be the responsibility of the client to ensure that the transaction is completed – like in a banking system, I envisage ways for the client to 'poll' the server to get the state of the last transaction, to make this possible.

|
| Operations such as filtering results are best done at the server unless
| you have extremely high-bandwidth connections so that each client can
| examine the entire data set and perform the operations themselves (with
| minimal time of course, since your other clients may be waiting on that
| transaction).

|
| Transactions, too, will have to be supported by the server, or else you
| may be left with partial transactions if a client gets disconnected
| somehow as well as the need to implement complex locking systems yourself.

|
| As for the other conditions, such as privilege and access control, I
| think you'd find that centrally managed is the most manageable in the
| long run.

|
| You won't regret making it server-centric. The experts have already done
| the optimisations and have ways of getting around the possible
| bottleneck of having a single server perform most of the operations.

|> |Yes, it's called a web frontend for a SQL DBMS. There's no shortage of
|> |Python frameworks to do this kind of things.

|> |

|>

|> I kind of wanted to avoid the web based stuff – the application data is so
small

|> and trivial, in a sense.

|
| You'd find that the python frameworks, especially those modeled after
| Ruby on Rails, make creating trivial applications, such as the front-end
| you describe, trivial. Go take a look at Django and TurboGears and the
| others. Some have videos demonstrating stuff like how to make a
| [blog/wiki/other-database-app in 5 minutes](#). Most come with a built-in
| webserver, though generally those aren't tested or guaranteed for
| high-load environments.

I get the feeling I am drinking out of a fire hose... I will look.

|
| All you need to add is a DB. Most recommend postgresql, and I'd
| recommend that too, to provide the features you are looking for. Avoid
| the lightweight DB systems such as Gadfly, sqlite, MS Jet/Access since
| those don't have the necessary features.

Re: Is there an obvious way to do this in python?

postgres seems the way to go – if there is anything that is coming across clear, it is this.

Have you any ideas about the "code change on the fly" requirement? or is it a complete no no?

thanks – Hendrik