

Deskripsi Service

SOAP

Return value pada service merupakan string berformat xml. Tag terluar (root) selalu <retval>. Misal:

```
<retval>
  <status>error</status>
  <desc>username sudah ada</desc>
  <other_attribute>some value</other_attribute>
</retval>
```

1. createUser(string username, string email, string nama_lengkap, string alamat, string provinsi, string kota, string kodepos, string telepon)
menambahkan user baru
return string berformat xml (atas):
 - <status> berisi `ok` jika sukses, `error` jika gagal
 - <desc> berisi error message, hanya ada jika status = `error`
 - <token> berisi string token login, akan disimpan di cookie / local storage. Hanya ada jika status ok
2. createBarang(string nama_barang, int harga, int stok, string kategori, string deskripsi)
menambahkan user baru
return string berformat xml (atas):
 - <status> berisi `ok` jika sukses, `error` jika gagal
 - <desc> berisi error message, hanya ada jika status = `error`
 - <id> berisi id barang yang baru ditambahkan, hanya ada jika status = `ok`

REST

Barang

Model data barang dalam bentuk json:

- id : integer id barang
- nama : string nama barang
- harga : integer harga
- stok : integer stok barang
- kategori : string kategori barang
- deskripsi : string deskripsi barang
- jumlah_beli : integer jumlah pembelian

1. GET barang/<id_barang>
mendapatkan barang dengan id = <id_barang>. error jika id tidak ada

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`
- barang : json representasi barang, hanya ada jika status = `ok`

2. GET kategori/<nama_kategori>

mendapatkan barang-barang dengan kategori = <nama_kategori>. error jika kategori tidak ada

GET parameter (url encoded):

- (opsional) page=<no_page> untuk nomor page. default=0
- (opsional) sort=name atau sort=harga. default=name
- (opsional) order=asc atau order=desc. default=asc

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`
- hasil : array of json representasi barang, hanya ada jika status = `ok`

3. GET search/<entry_search>

mencari barang dengan query = <entry_search>

GET parameter (url encoded):

- (opsional) page=<no_page> untuk nomor page.
- (opsional) sort=name atau sort=harga.
- (opsional) order=asc atau order=desc.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`
- hasil : array of json representasi barang, hanya ada jika status = `ok`

4. GET populer

mencari 10 barang dengan jumlah pembelian terbanyak

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`
- hasil : array of json representasi barang, hanya ada jika status = `ok`

5. PUT barang/<id_barang>

mengupdate data barang. Error jika token invalid / bukan token admin.

PUT data berformat json:

- barang : json representasi barang. Seluruh isi data, kecuali id, akan berubah
- token : string token login user.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

6. DELETE barang/<id_barang>

menghapus data barang. Error jika token invalid / bukan token admin.

DELETE data berformat JSON:

- ids : array of integer, id barang yang akan dihapus.
- token : string token login user.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

7. POST cart/<id_barang>

mengecek apakah barang cukup jika dimasukkan ke cart. Error jika barang tidak cukup atau token invalid.

POST data berformat JSON:

- jumlah: integer, jumlah barang yang ingin dimasukkan
- token : string token login user.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

8. POST buy

membeli barang. Error jika barang tidak mencukupi atau token invalid.

POST data berformat JSON:

- cart : json object dengan format {id_barang1 : jumlah1, id_barang2 : jumlah2, dst...}
- token : string token login user.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

User

Model data credit card dalam bentuk json:

- card_name : string nama kartu kredit
- card_number : string nomor kartu kredit
- card_date : string tanggal kadaluarsa kartu kredit

Model data user dalam bentuk json:

- username : string username
- password : string password user
- email : string email user
- nama_lengkap : string nama lengkap user
- alamat : string alamat user
- provinsi : string provinsi user
- kota : string kota user
- kodepos : string kode pos user
- telepon : string telepon user
- jumlah_transaksi : integer jumlah transaksi user
- has_card : boolean apakah user sudah mengisi info credit card atau tidak
- role : "admin" atau "user"

1. GET user/<username>

mendapatkan user dengan username = <username>. error jika user tidak ada atau token invalid.

GET parameter (url encoded):

- token = string token login user.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`
- user : json representasi user, kecuali password, hanya ada jika status = `ok`

2. GET user/<username>/card

mendapatkan apakah user sudah mendaftarkan card atau belum. error jika user tidak ada atau token invalid.

GET parameter (url encoded):

- token = string token login user.

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

- has_card : true/false, hanya ada jika status = `ok`

3. POST login

mengecek login. error jika username atau password salah.

POST data berformat JSON:

- username : string username
- password : string password

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`
- token: string token login, akan disimpan di cookie / local storage. Hanya ada jika status ok

4. POST logout

Logout.

POST data berformat JSON:

- username: string username
- token: string token login

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

5. PUT user/<username>

mengupdate data user dengan username = <username>. error jika username tidak ada atau token invalid.

PUT data berformat JSON:

- user: json representasi user.
- token: string token login user

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`

6. PUT user/<username>/card

mengupdate informasi kartu kredit user. error jika username tidak ada atau token invalid.

PUT data berformat JSON:

- card: json representasi credit card.

- token: string token login user

return string berformat json:

- status : `ok` jika sukses, `error` jika gagal
- desc : error message, hanya ada jika status = `error`