API Specification

Conventions

- **Frontend**- Front End server
- **Order** – Order Server
- **Catalog-** Catalog Server
- **Status** - HTTP status code of response.
- All the possible responses are listed under ‘Responses’ for each method. Only one of them is issued per request server.
- All response are in JSON format.
- All request parameters are mandatory unless explicitly marked as optional.

Status Codes

All status codes are standard HTTP status codes. The below ones are used in this API.

2XX - Success of some kind
4XX - Error occurred in client’s part
5XX - Error occurred in server’s part

Status Code Used	Description
200	OK
201	Created
400	Bad request
401	Unauthorized
422	Unprocessable Entity
404	Resource not found
500	Internal Server Error

Methods for the Front End Server

1. Buy

Call the buy method on the order server to buy an item with given ID

Request

Method	URL
GET	/buy

Query Parameters	Optional	Value Type	Description
id	False	String	The id of the book that needs to be bought

Example: localhost:8010/buy?id=1

Response

Status	Response

200	{
	"message": "Item with id 1 bought successfully.",
	"order": {
	"id": 196
	},
	"status": "Success",
	"validation_code": 200
	}

404	{
	"message": "The item with id 1 is no longer present in the catalog server",
	"status": "Failed",
	"validation_code": 404
	}

2. Search

Search for all the books belonging to a certain topic

Request

Method	URL
GET	/search

Query Parameters	Optional	Value Type	Description
topic	False	String	Topic of books to search for

Example localhost:8010/search?topic=distributed systems

Responses

Status	Response

200	{
	  "item": [
	  {
	  "cost": 10,
	  "count": 20,
	  "id": 1,
	  "title": "How to get a good grade in 677 in 20 minutes a day.",
	  "topic": "distributed systems"
	  },
	  {
	  "cost": 10,
	  "count": 15,
	  "id": 2,
	  "title": "RPCs for Dummies.",
	  "topic": "distributed systems"
	  }
	  ],
	  "message": {},

	<pre> "status": "Success", "validation_code": 200 }</pre>

3. Lookup

To lookup a particular item in the server

Request

Method	URL
GET	/lookup

Query Parameters	Optional	Value Type	Description
id	False	int	Id of the book to search for

Example - localhost:8010/lookup?id=3

Response

Status	Response

200	<pre>{
	  "item": [
	  {
	  "cost": 10,
	  "count": 0,
	  "id": 3,
	  "title": "Xen and the Art of Surviving Graduate School.",
	  "topic": "graduate school"
	  }
	  ],
	  "message": {},
	  "status": "Success",
	  "validation_code": 200
	}</pre>

Methods for the Catalog Server

1. Item

Look for Items in the database either by topic, ID, or return all books if no parameters are passed.

Request

Method	URL
GET	/item/<id_>
GET	/item

Query Parameters	Optional	Value Type	Description
id	True	String	The id of the book to look up
topic	True	String	Return all books with given topic

Example: localhost:8011/item?topic=distributed systems

Response

Status	Response
200	{ "item": [{ "cost": 10, "count": 20, "id": 1, "title": "How to get a good grade in 677 in 20 minutes a day.", "topic": "distributed systems" }, {

	"cost": 10, "count": 15, "id": 2, "title": "RPCs for Dummies.", "topic": "distributed systems" }], "message": {}, "status": "Success", "validation_code": 200 }

2. Update By ID

Update either the cost, or the count of the book corresponding to the given ID. Also propagates this update by calling the propagate by ID method of other replicas of catalog server.

Request

Method	URL
PUT	/item/<id_>

Query Parameters	Optional	Value Type	Description
id	False	String	The ID of the book whose count or cost needs to be updated. Count and cost must be passed as payload.

Responses

Status	Response
201	{ "item": {}, "message": {}, "status": "Success",

	"validation_code": 201 }

3. Propagate By ID

Propagate an update to either the cost, or the count of the book corresponding to the given ID

Request

Method	URL
PUT	/item/propagate/<id_>

Query Parameters	Optional	Value Type	Description
id	False	String	The ID of the book whose count or cost needs to be updated. Count and cost must be passed as payload.

Responses

Status	Response

201	{
	"item": {},
	"message": {},
	"status": "Success",
	"validation_code": 201
	}

Methods for the Order Server

1. Buy

Check if the book with given ID is available, and then buy it. Upon successful transaction, propagate the update to other replicas of the order server.

Request

Method	URL
GET	/buy/<item_id>

Query Parameters	Optional	Value Type	Description
id	False	String	The id of the book to buy

http://localhost:8012/buy/1

Response

Status	Response

200	{
	"message": "Item with id 1 bought successfully.",
	"order": {
	"id": 195
	},
	"status": "Success",
	"validation_code": 200
	}

404	{
	"message": "The item with id 1 is no longer present in the catalog server",
	"status": "Failed",
	"validation_code": 404
	}

2. Add to order

Propagate an update to the database of the order server. This ensures consistency between the databases of each replica of the order server

Request

Method	URL
PUT	/update

Responses

Status	Response

201	{
	"item": {},
	"message": {successfully updated order id 2},
	"status": "Success",
	"validation_code": 201
	}