MARKETPLACE (MP) CONCEPTS & GOALS

MarketPlace (MP) is an application to Create, Review, Update, and Delete Ah! Le Cart (ALC)'s inventory warehouses. MP is created and utilized across 3 programs to better assist in ALC endeavors:

MySQL Workbench - Where the database is located, and product information is logged.

IntelliJ: Where the application is written in Java to tell the application how to act. It links up with MySQL to assist in creating an organizational structure for ALC's inventory, while telling PostMan what information it should pull when a certain URL is typed in by the user.

PostMan: Where we test IntelliJ's URL paths to see if the user will get the right information when tasked.

GOAL: MP's goal is to make ALC's inventory management as simple as possible through the use of clear language and simple internet pathways. The application will allow ALC's employees to:

1. Create
   1. New products
   2. New product categories
   3. New Manufacturers
   4. New Invoices
2. Review
   1. Past Invoices
   2. Current Inventory Status by searchable properties (Like price, quantity, manufacturer name , etc)
3. Update
   1. Invoices
   2. Product Ratings
   3. Manufacturer Information
4. Delete
   1. Products no longer carried
   2. Product categories no longer utilized
   3. Manufacturers no longer utilized by ALC

**CURRENT COMMAND STATUS FOR ACCESSORIES**

* **Postman Set: What you should select before typing in the URL in Postman (this is typically Get, Set, Put, Post, Delete)**
* **Location in IntelliJ: Name of the class where you will find this code in IntelliJ**
* **Affects: What products/categories this code will interact with**
* **URL: What you will type in after the postman set is selected**
* **Example: Your code *should* look similar to the given example here.**
* **Working: Is this code currently operational?**
* **Notes: Additional information on the code**
* Postman Set: POST
* Location in IntelliJ: accessoriesController
* Affects: accessories - adds new product to table
* URL: <http://localhost:8080/accessories/>
* Example: <http://localhost:8080/accessories/>
* Working: No
* Notes: Currently displaying a 400 “bad request” error. Json cannot parse error: Cannot deserialize instance. Currently reviewing
* **Postman Set: GET**
* **Location in IntelliJ: accessoriesController**
* **Affects: accessories – shows user the full, current list of accessories**
* **URL:** [**http://localhost:8080/accessories/**](http://localhost:8080/accessories/)
* **Example:** [**http://localhost:8080/accessories/**](http://localhost:8080/accessories/)
* **Working: Yes**
* **Notes: No additional notes.**
* Postman Set: GET
* Location in IntelliJ: accessoriesController
* Affects: accessories – shows user a product based on the accessory product ID number typed in
* URL: http://localhost:8080/accessories/ID/{User’s typed in accessory ID number}
* Example: <http://localhost:8080/accessories/ID/1>
* Working: Yes
* Notes: No additional notes
* **Postman Set: GET**
* **Location in IntelliJ: accessoriesController**
* **Affects: accessories – shows user a product based on the accessory product Name typed in**
* **URL: http://localhost:8080/accessories/ID/{User’s typed in accessory Name}**
* **Example: http://localhost:8080/accessories/Name/Incense Pendant**
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: accessoriesController
* Affects: accessories – shows user list of products based on the accessory product price typed in
* URL: http://localhost:8080/accessories/Price/User’s typed in accessory price
* Example: <http://localhost:8080/accessories/Price/9.99>
* Working: Yes
* Notes: No additional notes
* **Postman Set: PUT**
* **Location in IntelliJ: accessoriesController**
* **Affects: accessories – updates an accessory table entry based on user’s input: accessory ID**
* **URL: http://localhost:8080/accessories/Update/User’s typed in accessory rating**
* **Example:** [**http://localhost:8080/accessories/Rate/1**](http://localhost:8080/accessories/Rate/1)
* **Working: No**
* **Notes: Currently showing an empty database as a response in Postman’s Body. Reviewing.**
* Postman Set: DELETE
* Location in IntelliJ: accessoriesController
* Affects: accessories – deletes an entry in the accessories table via user’s input: accessory ID
* URL: http://localhost:8080/accessories/Delete/User’s typed in accessory product ID number
* Example: <http://localhost:8080/accessories/Delete/1>
* Working: No
* Notes: Path currently shows 404 error in Postman Body with no additional information. Reviewing.

**CURRENT COMMAND STATUS FOR APPAREL**

* **Postman Set: What you should select before typing in the URL in Postman (this is typically Get, Set, Put, Post, Delete)**
* **Location in IntelliJ: Name of the class where you will find this code in IntelliJ**
* **Affects: What products/categories this code will interact with**
* **URL: What you will type in after the postman set is selected**
* **Example: Your code *should* look like the given example.**
* **Working: Is this code currently operational?**
* **Notes: Additional information on the code**
* Postman Set: POST
* Location in IntelliJ: apparelController
* Affects: apparel - adds new product to table
* URL: <http://localhost:8080/apparel/>
* Example: <http://localhost:8080/apparel/>
* Working: No
* Notes: Currently displaying a 400 “bad request” error. Json cannot parse error: Cannot deserialize instance. Currently reviewing
* **Postman Set: GET**
* **Location in IntelliJ: apparelController**
* **Affects: apparel – shows user the full, current list of apparel**
* **URL:** [**http://localhost:8080/apparel/**](http://localhost:8080/apparel/)
* **Example:** [**http://localhost:8080/apparel/**](http://localhost:8080/apparel/)
* **Working: Yes**
* **Notes: No additional notes.**
* Postman Set: GET
* Location in IntelliJ: apparelController
* Affects: apparel – shows user a product based on the apparel product ID number typed in
* URL: http://localhost:8080/apparel/id/{User’s typed in apparel ID number}
* Example: <http://localhost:8080/apparel/ID/1>
* Working: No
* Notes: Postman citing missing URI template. Reviewed, but currently cannot find the issue. Still reviewing
* **Postman Set: GET**
* **Location in IntelliJ: apparelController**
* **Affects: apparel – shows user a product based on the apparel product Name typed in**
* **URL: http://localhost:8080/apparel/name/{User’s typed in apparel name}**
* **Example: http://localhost:8080/accessories/Name/Plain White T**
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: apparelController
* Affects: apparel – shows user list of products based on the apparel product price typed in
* URL: http://localhost:8080/apparel/price/User’s typed in apparel price
* Example: <http://localhost:8080/apparel/price/74.99>
* Working: Yes
* Notes: No additional notes
* **Postman Set: PUT**
* **Location in IntelliJ: apparelController**
* **Affects: apparel – shows user list of products based on the apparel product price typed in**
* **URL: http://localhost:8080/apparel/Price/User’s typed in apparel rating**
* **Example:** [**http://localhost:8080/apparel/update/1**](http://localhost:8080/apparel/update/1)
* **Working: No**
* **Notes: Error 400; Bad Request – Json parse error: Cannot deserialize – reviewing**
* Postman Set: DELETE
* Location in IntelliJ: apparelController
* Affects: apparel – delete’s an entry in the apparel table by inputing Delete/apparel’s id number
* URL: http://localhost:8080/apparel/delete/User’s typed in apparel id number
* Example: <http://localhost:8080/apparel/delete/1/>
* Working: No
* Notes: Error 400; Bad Request – Json parse error: Cannot deserialize - reviewing

**CURRENT COMMAND STATUS FOR ELECTRONICS**

* **Postman Set: What you should select before typing in the URL in Postman (this is typically Get, Set, Put, Post, Delete)**
* **Location in IntelliJ: Name of the class where you will find this code in IntelliJ**
* **Affects: What products/categories this code will interact with**
* **URL: What you will type in after the postman set is selected**
* **Example: Your code *should* look like the given example.**
* **Working: Is this code currently operational?**
* **Notes: Additional information on the code**
* Postman Set: POST
* Location in IntelliJ: electronicsController
* Affects: electronics - adds new product to table
* URL: <http://localhost:8080/electronics/>
* Example: <http://localhost:8080/electronics/>
* Working: No
* Notes: Currently displaying a 400 “bad request” error. Json cannot parse error: Cannot deserialize instance. Currently reviewing – running trend with all the ID based commands
* **Postman Set: GET**
* **Location in IntelliJ: electronicsController**
* **Affects: electronics – shows user the full, current list of electronics**
* **URL:** [**http://localhost:8080/electronics/**](http://localhost:8080/electronics/)
* **Example:** [**http://localhost:8080/electronics/**](http://localhost:8080/electronics/)
* **Working: Yes**
* **Notes: No additional notes.**
* Postman Set: GET
* Location in IntelliJ: electronicsController
* Affects: electronics – shows user a product based on the electronics product ID number typed in
* URL: http://localhost:8080/electronics/id/{User’s typed in electronics ID number}
* Example: <http://localhost:8080/electronics/id/1>
* Working: Yes
* Notes: No additional notes
* **Postman Set: GET**
* **Location in IntelliJ: electronicsController**
* **Affects: electronics – shows user a product based on the electronics product Name typed in**
* **URL: http://localhost:8080/electronics/name/{User’s typed in electronics name}**
* **Example: http://localhost:8080/electronics/name/The Beast**
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: electronicsController
* Affects: electronics – shows user list of products based on the electronics product price typed in
* URL: http://localhost:8080/electronics/price/User’s typed in electronics price
* Example: <http://localhost:8080/electronics/price/499.99>
* Working: Yes
* Notes: No additional notes
* **Postman Set: PUT**
* **Location in IntelliJ: electronicsController**
* **Affects: electronics – shows user list of products via the electronics product price typed in**
* **URL: http://localhost:8080/electronics/update/User’s typed in electronics ID number**
* **Example:** [**http://localhost:8080/electronics/update/1**](http://localhost:8080/electronics/update/1)
* **Working: No**
* **Notes: Missing URI template for variable “elec” – need to review.**
* Postman Set: DELETE
* Location in IntelliJ: electronicsController
* Affects: electronics – delete’s an entry in the electronics table via electronics’s id number
* URL: http://localhost:8080/electronics/delete/User’s typed in electronics id number
* Example: <http://localhost:8080/electronics/delete/1/>
* Working: YES
* Notes: No additional Notes

**CURRENT COMMAND STATUS FOR BEAUTY**

* **Postman Set: What you should select before typing in the URL in Postman (this is typically Get, Set, Put, Post, Delete)**
* **Location in IntelliJ: Name of the class where you will find this code in IntelliJ**
* **Affects: What products/categories this code will interact with**
* **URL: What you will type in after the postman set is selected**
* **Example: Your code *should* look like the given example.**
* **Working: Is this code currently operational?**
* **Notes: Additional information on the code**
* Postman Set: POST
* Location in IntelliJ: beautyController
* Affects: Beauty - adds new product to table
* URL: <http://localhost:8080/beauty/>
* Example: <http://localhost:8080/beauty/>
* Working: No
* Notes: Currently displaying a 400 “bad request” error. Json cannot parse error: Cannot deserialize instance. Currently reviewing – running trend with all the ID based commands
* **Postman Set: GET**
* **Location in IntelliJ: beautyController**
* **Affects: beauty – shows user the full, current list of beauty**
* **URL:** [**http://localhost:8080/beauty/**](http://localhost:8080/beauty/)
* **Example:** [**http://localhost:8080/beauty/**](http://localhost:8080/beauty/)
* **Working: Yes**
* **Notes: No additional notes.**
* Postman Set: GET
* Location in IntelliJ: beautyController
* Affects: beauty – shows user a product based on the beauty product ID number typed in
* URL: http://localhost:8080/beauty/id/{User’s typed in beauty ID number}
* Example: <http://localhost:8080/beauty/id/1>
* Working: Yes
* Notes: No additional notes
* **Postman Set: GET**
* **Location in IntelliJ: beautyController**
* **Affects: beauty – shows user a product based on the beauty product Name typed in**
* **URL: http://localhost:8080/beauty/name/{User’s typed in beauty name}**
* **Example: http://localhost:8080/beauty/name/Noire Red Lipstick**
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: beautyController
* Affects: beauty – shows user list of products based on the beauty product price typed in
* URL: http://localhost:8080/beauty/price/User’s typed in beauty’s price
* Example: <http://localhost:8080/beauty/price/14.99>
* Working: Yes
* Notes: No additional notes
* **Postman Set: GET**
* **Location in IntelliJ: beautyController**
* **Affects: beauty – shows user list of products based on the beauty product quantity typed in**
* **URL: http://localhost:8080/beauty/price/User’s typed in beauty’s price**
* **Example:** [**http://localhost:8080/beauty/price/14.99**](http://localhost:8080/beauty/price/14.99)
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: beautyController
* Affects: beauty – shows user list of products based on the beauty product price typed in
* URL: http://localhost:8080/beauty/rating/User’s typed in beauty product rating
* Example: <http://localhost:8080/beauty/rating/5>
* Working: Yes
* Notes: No additional notes
* **Postman Set: PUT**
* **Location in IntelliJ: beautyController**
* **Affects: beauty – updates a beauty table entry via ID**
* **URL: http://localhost:8080/beauty/update/User’s typed in beauty ID**
* **Example:** [**http://localhost:8080/beauty/update/1**](http://localhost:8080/beauty/update/14.99)
* **Working: No**
* **Notes: Error 400; Bad Request – Json parse error: Cannot deserialize – reviewing**
* Postman Set: DELETE
* Location in IntelliJ: beautyController
* Affects: beauty – deletes a beauty table entry via ID
* URL: http://localhost:8080/beauty/delete/User’s typed in beauty ID
* Example: <http://localhost:8080/beauty/delete/1>
* Working: Yes
* Notes: No additional notes needed.

**CURRENT COMMAND STATUS FOR GARDEN**

* **Postman Set: What you should select before typing in the URL in Postman (this is typically Get, Set, Put, Post, Delete)**
* **Location in IntelliJ: Name of the class where you will find this code in IntelliJ**
* **Affects: What products/categories this code will interact with**
* **URL: What you will type in after the postman set is selected**
* **Example: Your code *should* look like the given example.**
* **Working: Is this code currently operational?**
* **Notes: Additional information on the code**
* Postman Set: POST
* Location in IntelliJ: gardenController
* Affects: garden - adds new product to table
* URL: <http://localhost:8080/garden/>
* Example: <http://localhost:8080/garden/>
* Working: No
* Notes: Currently displaying a 400 “bad request” error. Json cannot parse error: Cannot deserialize instance. Currently reviewing – running trend with all the ID based commands
* **Postman Set: GET**
* **Location in IntelliJ: gardenController**
* **Affects: garden – shows user the full, current list of garden products**
* **URL:** [**http://localhost:8080/garden/**](http://localhost:8080/garden/)
* **Example:** [**http://localhost:8080/garden/**](http://localhost:8080/garden/)
* **Working: Yes**
* **Notes: No additional notes.**
* Postman Set: GET
* Location in IntelliJ: gardenController
* Affects: garden – shows user a product based on the beauty product ID number typed in
* URL: http://localhost:8080/beauty/id/{User’s typed in beauty ID number}
* Example: <http://localhost:8080/beauty/id/1>
* Working: Yes
* Notes: No additional notes
* **Postman Set: GET**
* **Location in IntelliJ: gardenController**
* **Affects: garden – shows user a product based on the garden product Name typed in**
* **URL: http://localhost:8080/garden/name/{User’s typed in garden name}**
* **Example: http://localhost:8080/garden/name/Bamboo Bench**
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: gardenController
* Affects: garden – shows user list of products based on the garden product price typed in
* URL: http://localhost:8080/garden/price/User’s typed in garden’s price
* Example: <http://localhost:8080/garden/price/24.99>
* Working: Yes
* Notes: No additional notes
* **Postman Set: GET**
* **Location in IntelliJ: gardenController**
* **Affects: garden – shows user list of products based on the garden product quantity typed in**
* **URL: http://localhost:8080/garden/quantity/User’s typed in beauty’s quantity**
* **Example: http://localhost:8080/beauty/quantity/30**
* **Working: Yes**
* **Notes: No additional notes**
* Postman Set: GET
* Location in IntelliJ: gardenController
* Affects: garden – shows user list of products based on the garden product rating typed in
* URL: http://localhost:8080/beauty/rating/User’s typed in garden product rating
* Example: <http://localhost:8080/garden/rate/5>
* Working: Yes
* Notes: No additional notes
* Postman Set: PUT
* Location in IntelliJ: gardenController
* Affects: garden – updates a garden table entry via ID
* URL: http://localhost:8080/garden/update/User’s typed in beauty ID
* Example: <http://localhost:8080/garden/update/1>
* Working: No
* Notes: Missing URI template for variable “gar” – need to review
* Postman Set: DELETE
* Location in IntelliJ: gardenController
* Affects: beauty – deletes a garden table entry via ID
* URL: http://localhost:8080/garden/delete/User’s typed in garden ID
* Example: <http://localhost:8080/garden/delete/1>
* Working: Yes
* Notes: No additional notes needed.