users\_api

1. Dependencies:
   1. pip install Flask (versión 3.1.0)
2. For testing use postman or any other similar
3. REST API listening in <http://localhost:5000>
4. END POINTS:
   1. / -------- home page:

* Use GET method
* Return Body HTML <h1>Rocky Pet Shop!</h1>
  1. / users -------- show the users:
* Use GET method
* Use query parameter (optional), show users by ‘type’, administrator or customer
* Returns:
  1. users list, 200 OK
  2. 400 Bad Request , If the query parameter ‘type’ is incorrect
  3. 404 File Not Found , If users.json does not exists
  4. 500 Internal Server Error, for any other error
  5. / users /<user\_id> -------- show a specific user:
     + Use GET method
     + Path parameter ‘user\_id’ identifier for only one and specific user
* Returns:
  1. user, 200 OK
  2. 400 Bad Request , If the user\_id does not exist
  3. 404 File Not Found , If users.json does not exists
  4. 500, for any other error
  5. / users -------- save user:
* Use POST method
* Body JSON:

{

"user\_id": "abc123",

"password": "secret",

"type": "customer",

"name": "Juan Blanco",

"email": "juan@example.com",

"address": "Calle Fallas"

}

* Validations:

1. All attributes are obligatory
2. ‘user\_id’ can’t be repeat
3. type must be ‘administrator’ or ‘customer’

* Returns:
  1. ‘Successful saved’, 200 OK
  2. 400 Bad Request, If user info is incomplete or wrong
  3. 500 Internal Server Error, for any other error
  4. / user/<user\_id> -------- update user:
* Use PATCH method
* Path parameter: ‘user\_id’
* Body JSON: \*\*\*one or more attributes can be skipped\*\*\*

{

"user\_id": "abc123",

"password": "secret",

"type": "customer",

"name": "Juan Blanco",

"email": "juan@example.com",

"address": "Calle Fallas"

}

* Validations:

1. correct ‘user\_id’
2. ‘type’ still has to be ‘administrator’ or ‘customer’

* Returns:
  1. user, 200 OK
  2. 400 Bad Request , If user id does not exist
  3. 400 Bad Request, If ‘type’ is wrong
  4. 404 File Not Found, If users.json does not exist
  5. 500 Internal Server Error, for any other error
  6. / users /<user\_id> -------- delete a specific user:
     + Use DELETE method
     + Path parameter ‘user\_id’ identifier for only one and specific user
* Validations:

1. correct ‘user\_id’

* Returns:
  1. “Successful delete”, users list, 200 OK
  2. 400 Bad Request , If user id does not exist
  3. 404 File Not Found, if users.json file does not exist
  4. 500 Internal Server Error, for any other error

products\_api

1. Dependencies:
   1. pip install Flask (versión 3.1.0)
2. For testing use postman or any other similar
3. REST API listening in <http://localhost:5000>
4. END POINTS:
5. / -------- home page:

* Use GET method
* Return Body HTML <h1> Rocky Pet Shop Products!</h1>

1. / products -------- show the products:

* Use GET method
* Use query parameter (optional), show products by ‘category’ ("food", "toys", "pharmacy", "accessories" or by ‘target species’ (dog, cat, fish, birds, etc.)
* Returns:

1. products list, 200 OK
2. 400 Bad Request, If the query parameter category of target species is incorrect
3. 404 File Not Found, If products.json does not exists
4. 500 Internal Server Error, for any other error
5. / products /<products\_id> -------- show a specific product:
   * + Use GET method
     + Path parameter ‘product\_id’ identifier for only one and specific product

* Returns:

1. product, 200 OK
2. 400 Bad Request, If the product\_id does not exist
3. 404 File Not Found, If products.json does not exists
4. 500, for any other error
5. / products -------- save product:

* Use POST method
* Body JSON:

{

"product\_id": "P001",

"name": "Canine Digestive Kibble",

"category": "food",

"description": "Easily digestible dry food for adult dogs.",

"target\_species": "Dog",

"supplier": "NutriPet",

"stock": 80,

"cost": 7.25,

"price": 15.99

}

* Validations:

1. All attributes are obligatory
2. ‘product\_id’ can’t be repeat
3. ‘category’ must be “food", "toys", "pharmacy", "accessories"
4. ‘stock’ must be an integer
5. ‘cost’ and ‘price’ must be float

* Returns:

1. ‘Successful saved’, 200 OK
2. 400 Bad Request, If ‘product id’ is repeat
3. 400 Bad Request, If ‘category’ is wrong
4. 400 Bad Request, If ‘stock’ has a wrong type
5. 400 Bad Request, If ‘cost’ or ‘price’ has a wrong type
6. 500 Internal Server Error, for any other error
7. / products/<product\_id> -------- update product:

* Use PATCH method
* Path parameter: ‘product\_id’
* Body JSON: \*\*\*one or more attributes can be skipped\*\*\*

{

"product\_id": "P001",

"name": "Canine Digestive Kibble",

"category": "food",

"description": "Easily digestible dry food for adult dogs.",

"target\_species": "Dog",

"supplier": "NutriPet",

"stock": 80,

"cost": 7.25,

"price": 15.99

}

* Validations:

1. correct ‘product\_id’
2. ‘category’ must be “food", "toys", "pharmacy", "accessories"
3. ‘stock’ must be an integer
4. ‘cost’ and ‘price’ must be float

* Returns:

1. ‘Successful updated’, product, 200 OK
2. 400 Bad Request, If ‘product id’ does not exist
3. 400 Bad Request, If ‘category’ is wrong
4. 400 Bad Request, If ‘stock’ has a wrong type
5. 400 Bad Request, If ‘cost’ or ‘price’ has a wrong type
6. 404 File Not Found, If products.json does not exist
7. 500 Internal Server Error, for any other error
8. / products/<product\_id> -------- delete a specific product:
   * + Use DELETE method
     + Path parameter ‘product\_id’ identifier for only one and specific user

* Validations:

1. correct ‘product\_id’

* Returns:

1. “Successful delete”, products list, 200 OK
2. 400 Bad Request , If product id does not exist
3. 404 File Not Found, if products.json file does not exist
4. 500 Internal Server Error, for any other error

invoices\_api

1. Dependencies:

* 1. pip install Flask (versión 3.1.0)

2. For testing use postman or any other similar

3. REST API listening in <http://localhost:5000>

4 . END POINTS:

1. / -------- home page:

* Use GET method
* Return Body HTML <h1> Invoice Control</h1>

1. / invoices -------- show the invoices:

* Use GET method
* Use query parameter (optional), show products by ‘email’ or shipping method "Standard", "Express", "Overnight"
* Returns:

1. Invoices list, 200 OK
2. 400 Bad Request, If the query parameter ‘email’ or ‘shipping\_method’ are wrong
3. 404 File Not Found, If invocies.json does not exists
4. 500 Internal Server Error, for any other error
5. / invoices/<invoice\_id> -------- show a specific invoice:
   * + Use GET method
     + Path parameter ‘invoice\_id’ identifier for only one and specific invoice

* Returns:

1. invoice, 200 OK
2. 400 Bad Request, If the invoice\_id does not exist or if is wrong
3. 404 File Not Found, If invoices.json does not exists
4. 500, for any other error
5. / invoices -------- save invoice:

* Use POST method
* Body JSON:

{

"invoice\_id": "prueba22",

"date\_placed": "06/05/2025 02:30 PM",

"shipping\_method": "Express",

"shipping\_address": {

"address": "San Francisco de dos rios, av 10, house# 501",

"canton": "Central",

"province": "San Jose",

"postal\_code": "11111"

},

"phone\_number": "8555-1234",

"email": "rugalde@example.com",

"payment\_method": "card",

"items": [

{

"product\_id": "d201",

"product\_name": "Ball toy",

"quantity": 2,

"price": 41.98,

"discount": 0.0,

"sales\_tax": 3.8,

"shipping\_cost": 8.0,

"total": 53.78

},

{

"product\_id": "c301",

"product\_name": "Fipronil Spot-On 3ml",

"quantity": 1,

"price": 19.99,

"discount": 2.0,

"sales\_tax": 1.17,

"shipping\_cost": 4.0,

"total": 23.16

}

],

"invoice\_total": 76.94

}

* Validations:

1. All attributes are obligatory
2. ‘invoice\_id’ can’t be repeat
3. ‘shipping\_method’ must be "Standard", "Express", "Overnight"
4. 'payment\_method’ must be debit card", "credit card", "SINPE", "bank transfer”
5. ‘quantity’ must be an integer
6. "price", "discount", "sales\_tax", "shipping\_cost", "total" and “invoice\_total” must be float

* Returns:

1. ‘Successful saved’, invoices list, 200 OK
2. 400 Bad Request, If ‘product id’ is repeat
3. 400 Bad Request, If ‘shipping\_method’ is wrong
4. 400 Bad Request, If ‘payment\_method’ is wrong
5. 400 Bad Request, If ‘quantity’ has a wrong type
6. 400 Bad Request, If ‘"price", "discount", "sales\_tax", "shipping\_cost", "total" or “invoice\_total” has a wrong type
7. 500 Internal Server Error, for any other error
8. / invoices/<invoice\_id> -------- update invoices:

* Use PATCH method
* Path parameter: ‘invoice\_id’
* Body JSON: \*\*\*one or more attributes can be skipped\*\*\*

\*\*\*However, if the shipping address dictionary needs to be update, all attributes of the shipping address must remain whether with new or the same information; the update can’t skip any attribute of it.\*\*\* Same happens with the items list (each item is a dictionary), whether with new or the same information, the update can't skip any attribute neither.\*\*\*If you don’t need to update shipping address or any item, feel free to skip the whole attribute without any problem.\*\*\*

{

"invoice\_id": "prueba22",

"date\_placed": "06/05/2025 02:30 PM",

"shipping\_method": "Express",

"shipping\_address": {

"address": "San Francisco de dos rios, av 10, house# 501",

"canton": "Central",

"province": "San Jose",

"postal\_code": "11111"

},

"phone\_number": "8555-1234",

"email": "rugalde@example.com",

"payment\_method": "card",

"items": [

{

"product\_id": "d201",

"product\_name": "Ball toy",

"quantity": 2,

"price": 41.98,

"discount": 0.0,

"sales\_tax": 3.8,

"shipping\_cost": 8.0,

"total": 53.78

},

{

"product\_id": "c301",

"product\_name": "Fipronil Spot-On 3ml",

"quantity": 1,

"price": 19.99,

"discount": 2.0,

"sales\_tax": 1.17,

"shipping\_cost": 4.0,

"total": 23.16

}

],

"invoice\_total": 76.94

}

* Validations:

1. correct ‘invoice\_id’
2. ‘shipping\_method’ must be "Standard", "Express", "Overnight"
3. 'payment\_method’ must be debit card", "credit card", "SINPE", "bank transfer”
4. ‘quantity’ must be an integer
5. "price", "discount", "sales\_tax", "shipping\_cost", "total" and “invoice\_total” must be float

* Returns:

1. ‘Successful updated’, invoice, 200 OK
2. 400 Bad Request, If ‘invoice id’ is does not exist or is wrong
3. 400 Bad Request, If ‘shipping\_method’ is wrong
4. 400 Bad Request, If ‘payment\_method’ is wrong
5. 400 Bad Request, If ‘quantity’ has a wrong type
6. 400 Bad Request, If ‘"price", "discount", "sales\_tax", "shipping\_cost", "total" or “invoice\_total” has a wrong type
7. 404 File Not Found, If invoices.json does not exist
8. 500 Internal Server Error, for any other error
9. / invoices/<invoice\_id> -------- delete a specific invoice:
   * + Use DELETE method
     + Path parameter ‘invoice\_id’ identifier for only one and specific user

* Validations:

1. correct ‘product\_id’

* Returns:

1. “Successful delete”, products list, 200 OK
2. 400 Bad Request , If invoice id does not exist
3. 404 File Not Found, if invoices.json file does not exist
4. 500 Internal Server Error, for any other error

singup\_login\_api

1. Dependencies:

a. pip install Flask (versión 3.1.0)

2. For testing use postman or any other similar

3. REST API listening in <http://localhost:5000>

4 . END POINTS:

1. / register -------- register an user:

* Use POST method
* The registration logic remains pending for future developments
* Return ‘message: Successful registration’, 201

1. /login -------- login an user:
   * Use POST method
   * The login logic remains pending for future developments
   * Return ‘message: Successful login’, 200