

API prototyping

Mock-based prototyping in WireMock Cloud

Simplify the API design process, shorten feedback cycles, and increase collaboration

GET https://openapi-demo.wiremockapi.cloud/posts	/{id}			
Gets a new post by ID		- Parameters		
Request		id* : integer detail* : full		
Path Parameters		tracking : string x-transaction-id* : string		
id integer	required	Send API Request		
Query Parameters		Request Sample: Java / AsyncHttp ~	t ⁶	
tracking string detail string Allowed values: full summary	required	<pre>AsyncHttpClient client = new DefaultAsyncHttpClie client.prepare("GET", "https://openapi-demo.wirem .setHeader("x-transaction-id", "") .setHeader("Accept", "application/json") .execute() .toCompletableFuture()</pre>		
Headers		<pre>.thenAccept(System.out::println) .join();</pre>		
x-transaction-id string	required	client.close();		



The Solution

Designing APIs is often inefficient and slow: complex specs need to be manually written based on murky requirements, details are often poorly defined or documented, and many problems only appear during implementation. This results in slow feedback loops, stalled development, and far too many 'back to the drawing board' moments. WireMock Cloud increases developer efficiency and replaces tedious iteration cycles with a collaborative API prototyping and testing environment. Here's how it works:

- **1. Start with a mock API, we'll do the rest.** WireMock Cloud will auto-generate the OpenAPI spec and relevant documentation in a collaborative portal
- 2. Continue developing in parallel. When the spec changes based on ongoing feedback or evolving business requirements – you can easily reflect the changes in the mock.
- **3. Go live without a hitch.** When the live API is ready, replace a few lines in your code and move to production – safe in the knowledge that your app has been developed and tested against the real API's expected behaviors.

"WireMock Prototyping module enables additional coverage for contract testing and better collaboration between team members"

Kosta Krauth

Head of Engineering, Bilt Rewards

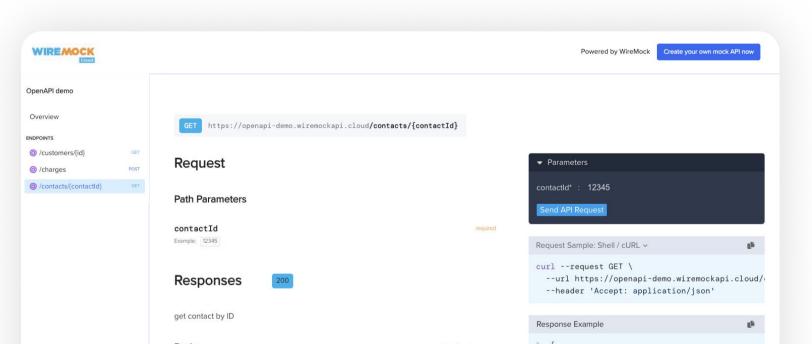
Open API example 🛛 🔒 🛚	share J API template					
https://m65v3.wiremockapi.cloud/ 📋						
+ New 🤤 Refresh Filter stub list						
Gets a customer by ID no ID in response	Request					
GET /customers/{id}	Name					
200	Gets a customer by ID no ID in response					
{ "lastName" : "Akehurst" }	Add description					
Add description						
Create new charge	GET ~ /customers/(id)					
POST /charges						
200	✓Advanced [®]					
{ "id": "9b6b418a-0b59-486c-a4a3-cd0flcdbf0b3"	URL match type Priority					
	Path template v 3 v					
Gets a customer by ID example-one						
GET /customers/{id}	O Query param Image: O Body Imag					
200	detail 🗆 NOT equalTo 🗸 full 🙆 (query)					
{ "id" : "5931ed73-dcd4-4e24-ad24-975ac288fa7f",	id v NOT equalTo v 5931ed73-dcd4-4e24-ad24-975ac288					
Create new charge - generic	id v INOT equalTo v 5931ed73-dcd4-4e24-ad24-975ac288 (path parameter)					
POST /charges	> Scenario ①					
200	/ scenario					
{ "id": "9b6b418a-0b59-406c-a4a3-cd0f1cdbf0b3"						
	Response					
Gets a customer by ID generic	Direct Fault Proxy					
GET /customers/{id}						
200	Status					
{ "id" : "5931ed73-dcd4-4e24-ad24-975ac288fa7f",	200 Enable dynamic response templating					

Increase collaboration and accelerate feedback loops

- Ground design discussions in working mock APIs that can be integrationtested
- Iterate on specs rapidly based on implementation feedback
- Improve governance with better documentation and adherence to OpenAPI standards

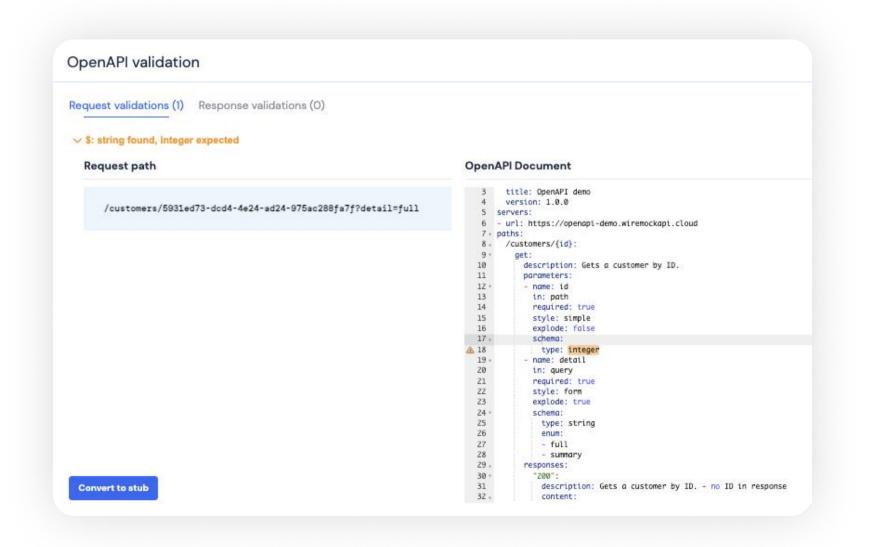
Streamline API development and increase efficiency in your API work

- Auto-generate OpenAPI specs and documentation at the earliest stage of your API development process
- Manage mocks and specs through a collaborative web UI to increase consistency and transparency



- Keep mocks and APIs up to date in your developer portal

Body	application/json 🗸	1 { 2 "id": 12345, 3 "name": "Tom"
id integer <int32> Example: 12345</int32>	required	4 }
name string	required	



Continuously validate your mock API against the OpenAPI spec for simplified contract testing

- Get alerts when your mocks receive requests or return responses that conflict the OpenAPI spec
- Identify issues and miscommunication earlier in the development lifecycle
- Keep your mock APIs close to the real APIs to ensure a seamless transition to production