Looker

Building Data Applications with the Looker Extension Framework

The Looker Extension Framework is a development framework that significantly reduces the effort and complexity of building custom data applications and tools, such as:

- Internal platform applications for your company
- External platforms for your customers, such as customer portals for Embedded Analytics applications built with data in Looker
- Targeted internal tools
- Applications for embedding in external applications

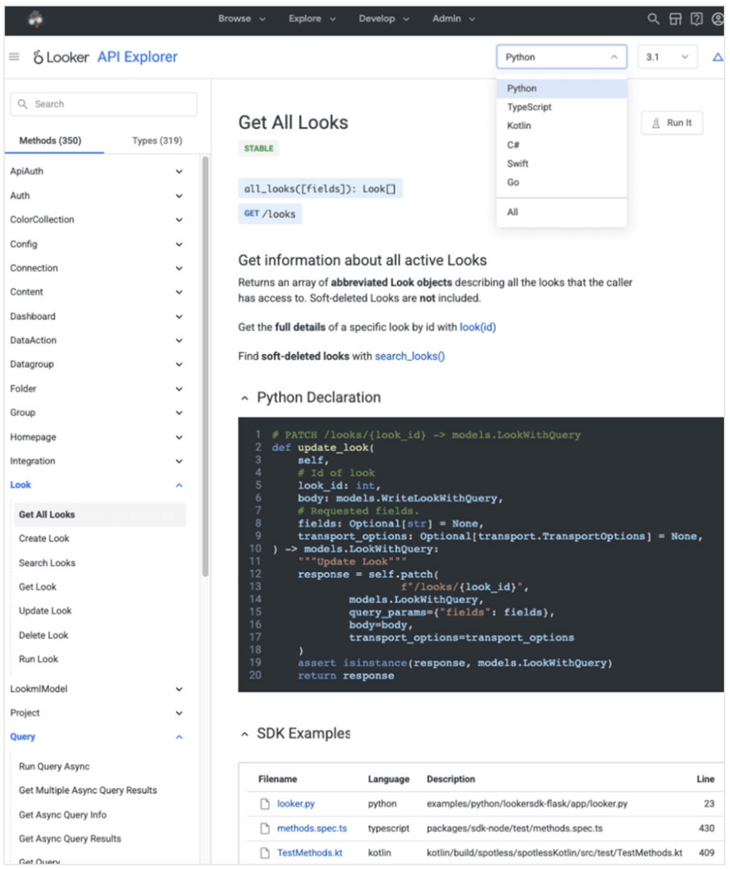
The Extension Framework takes care of some of the more tedious aspects of building a web application so that you can focus on starting development right away. Custom applications and tools created with the Extension Framework can be accessed from within Looker, allowing the Extension Framework to handle:

- Hosting
- Authentication
- Authorization
- API access, including letting you leverage other common developer resources, such as third-party API endpoints, within Looker

Developers have used the Extension Framework to build innovative and useful applications available in the Looker marketplace. Let's look at three examples of such applications.

API Explorer for Discovering More within the Looker API

API Explorer is a new, interactive, and easy way to explore the Looker API. It helps you prototype requests in the language of your choice, and actually execute API calls.



Get All Looks

STABLE

`all_looks([fields]): Look[]`

`GET /looks`

Get information about all active Looks

Returns an array of **abbreviated Look objects** describing all the looks that the caller has access to. Soft-deleted Looks are **not** included.

Get the **full details** of a specific look by id with look(id)

Find **soft-deleted looks** with search_looks()

Python Declaration

```python
# PATCH /looks/{look_id} -> models.LookWithQuery
def update_look(
    self,
    # Id of look
    look_id: int,
    body: models.WriteLookWithQuery,
    # Requested fields.
    fields: Optional[str] = None,
    transport_options: Optional[transport.TransportOptions] = None,
) -> models.LookWithQuery:
    """Update Look"""
    response = self.patch(
                    f"/looks/{look_id}",
            models.LookWithQuery,
            query_params={"fields": fields},
            body=body,
            transport_options=transport_options
    )
    assert isinstance(response, models.LookWithQuery)
    return response
```

SDK Examples

Filename	Language	Description	Line
looker.py	python	examples/python/lookersdk-flask/app/looker.py	23
methods.spec.ts	typescript	packages/sdk-node/test/methods.spec.ts	430
TestMethods.kt	kotlin	kotlin/build/spotless/spotlessKotlin/src/test/TestMethods.kt	409

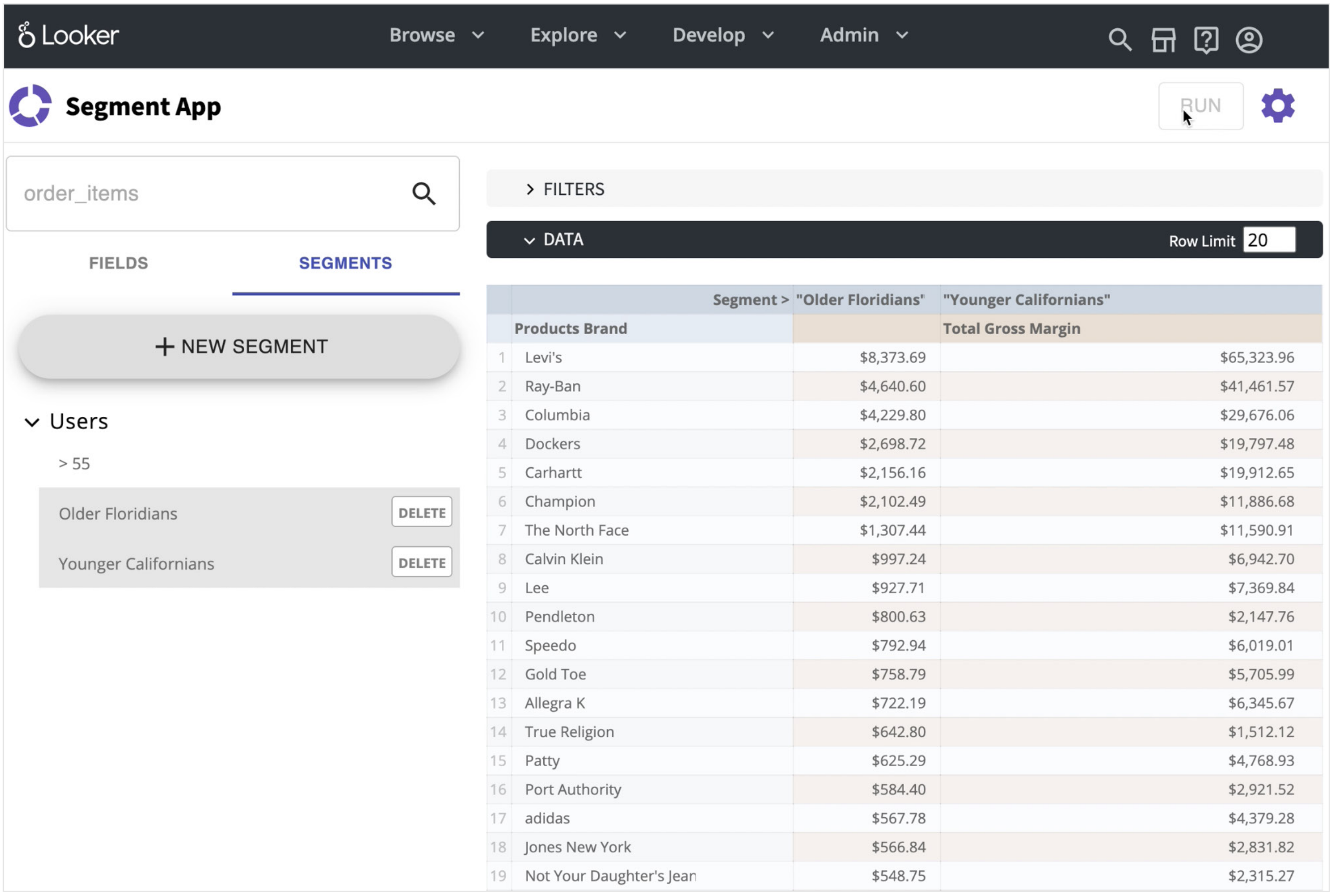
The API Explorer provides all the details you expect from documentation and then goes beyond that to help developers get from Zero to 'Hello World' as fast as possible.

'Run It' lets you execute complex API calls without writing any code. You provide your inputs in a hard-to-mess-up form UI (rather than complex dicts or models), press the button, and then it runs your call. The response comes in a standard JSON format, but can also return complex formats such as PDFs or PNGs.

'Example mining' lets you immediately see in-context examples of the function you're interested in, and may even link you to an example script that does exactly what you're hoping to accomplish. If you're interested in all the features, install it from the Marketplace and explore on your own.

Segment by 4 Mile Analytics for Dataset Metrics Comparisons

The Segment application, built by 4 Mile Analytics, enables Looker users to define, save, and apply a collection of filter criteria "segments" to any Looker query. Segments allow easy comparison of metrics between slices of a dataset, answering questions like, "How does the conversion rate of Texas millennials compare to the conversion rate of New York baby boomers, and how do they compare to the total user base?"



	Segment >	"Older Floridians"	"Younger Californians"
	Products Brand	Total Gross Margin	
1	Levi's	$8,373.69	$65,323.96
2	Ray-Ban	$4,640.60	$41,461.57
3	Columbia	$4,229.80	$29,676.06
4	Dockers	$2,698.72	$19,797.48
5	Carhartt	$2,156.16	$19,912.65
6	Champion	$2,102.49	$11,886.68
7	The North Face	$1,307.44	$11,590.91
8	Calvin Klein	$997.24	$6,942.70
9	Lee	$927.71	$7,369.84
10	Pendleton	$800.63	$2,147.76
11	Speedo	$792.94	$6,019.01
12	Gold Toe	$758.79	$5,705.99
13	Allegra K	$722.19	$6,345.67
14	True Religion	$642.80	$1,512.12
15	Patty	$625.29	$4,768.93
16	Port Authority	$584.40	$2,921.52
17	adidas	$567.78	$4,379.28
18	Jones New York	$566.84	$2,831.82
19	Not Your Daughter's Jean	$548.75	$2,315.27

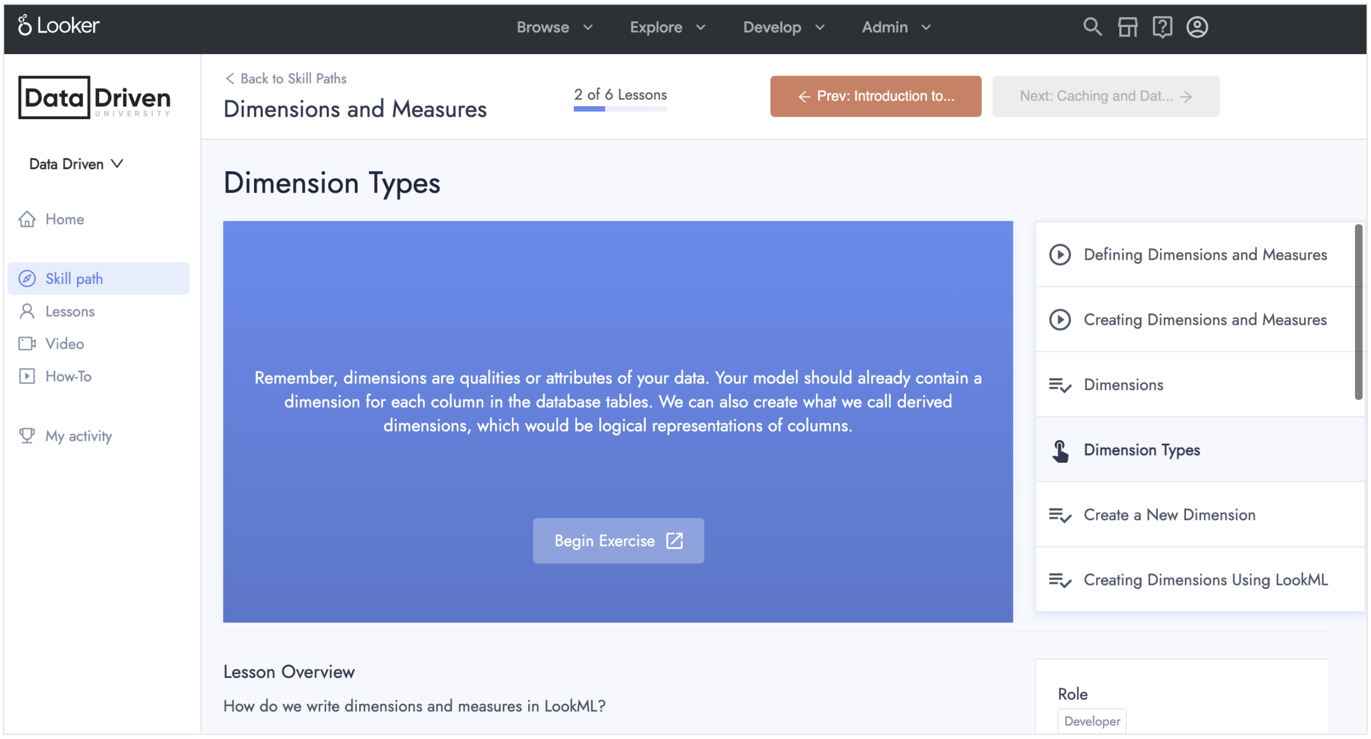
Extension Framework applications can be quickly developed and deployed as part of Looker's core codebase. Because the applications are hosted inside Looker, users are already authenticated and can visualize their model without leaving Looker. Looker UI Components further enable application developers to match the look and feel of the Looker platform, accelerating front-end development.

Data Driven University for Online Learning

Data Driven University is an online learning platform that leverages a Chrome extension to allow users to 'learn by doing' when completing training exercises directly in Looker. It helps customers quickly onboard business users, developers, and admins to get ROI on the product. Data Driven had an existing and separate Node.js application, but wanted to create an Extension Framework version of their application so they could meet their users where they already lived—inside Looker!

One developer took one day to port their application to the Looker Extension Framework, which allows rapid migration because it uses industry standard technologies. Very minimal changes were required to their codebase, and the open source example repositories and documentation were extremely useful in giving this company the blueprint for migration. Another huge win for them was the ability to take advantage of the Extension SDK: it provided built-in methods for storing secrets and securely connecting to their backend.

"Developing against the Looker Extension Framework was a huge win for us. We were able to delight our users by providing them with an integrated experience inside of Looker. Our development team was able to quickly migrate our application while maintaining privacy and security. We've delivered a tremendous amount of value to our shared users with minimal lift on our end." Jawad Laraqui. CEO, Data Driven.



Dimensions and Measures

Dimension Types

Remember, dimensions are qualities or attributes of your data. Your model should already contain a dimension for each column in the database tables. We can also create what we call derived dimensions, which would be logical representations of columns.

Lesson Overview

How do we write dimensions and measures in LookML?

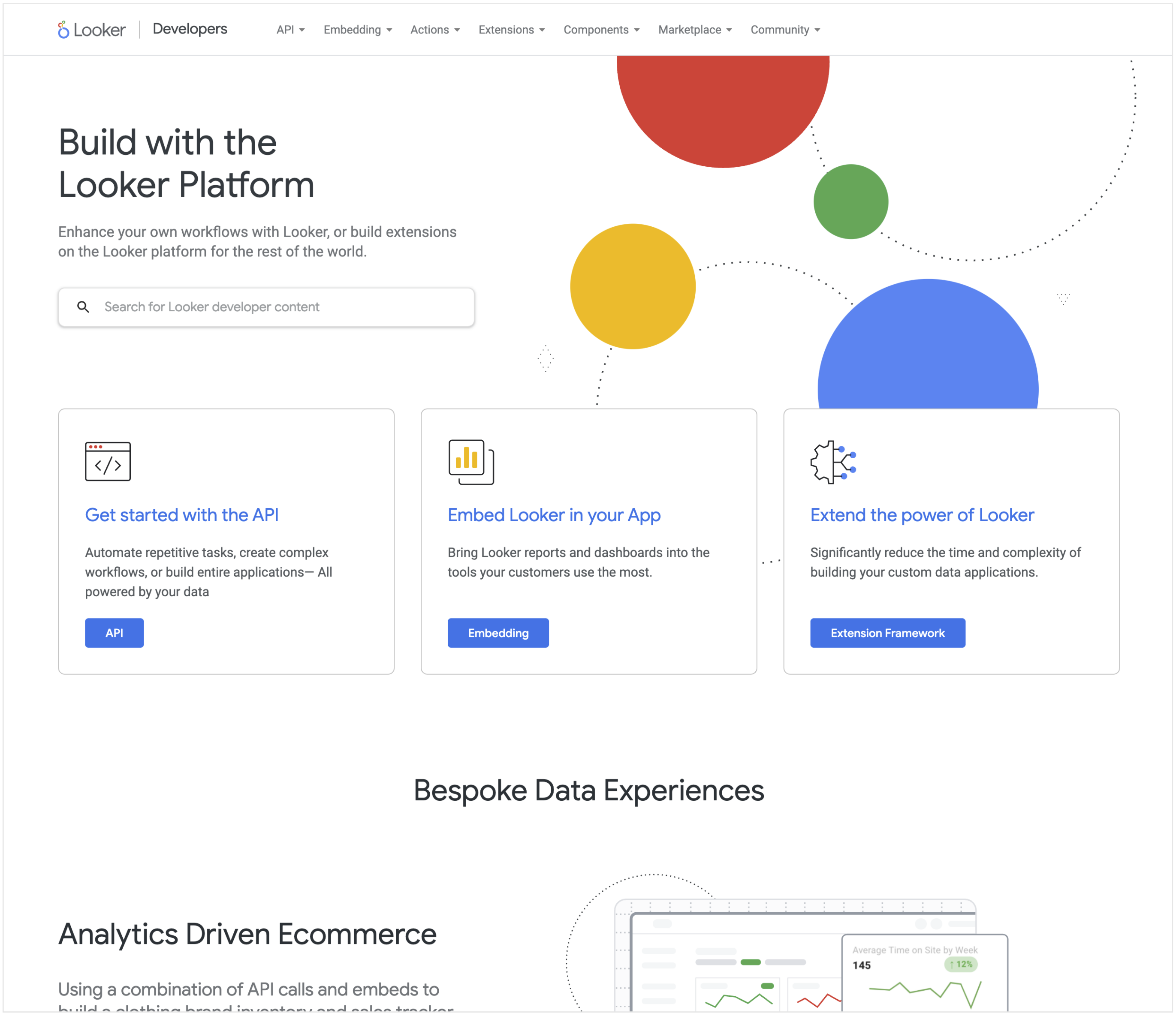
Start Building with the Extension Framework

These three examples demonstrate the breadth and depth of applications developers have built using the Extension Framework.

The next app made with the Extension Framework could very well be one that you create.

The easiest way to get started is to first generate a new starter extension from a template, and then customize and add functionality to that starter. This ensures that all configuration and packaging is correct, which can be difficult to do manually.

See example on the Looker Developer Portal on how to create a new Looker project for your extension and generate a starter extension.



Build with the Looker Platform

Enhance your own workflows with Looker, or build extensions on the Looker platform for the rest of the world.

Search for Looker developer content

Get started with the API

Automate repetitive tasks, create complex workflows, or build entire applications— All powered by your data

API

Embed Looker in your App

Bring Looker reports and dashboards into the tools your customers use the most.

Embedding

Extend the power of Looker

Significantly reduce the time and complexity of building your custom data applications.

Extension Framework

Bespoke Data Experiences

Analytics Driven Ecommerce

Using a combination of API calls and embeds to