Couchbase

Architecture and Administration Basics

Workshop **Day 2 – Labs**
C/C++

Davis Chapman, Solution Architect
https://github.com/couchbase-ps/cb-workshop-2d/

1 | Installation & Configuration SDK

SDK C - Sample Application

We will be using the sample application for this workshop, the description of which can be found:

- https://docs.couchbase.com/c-sdk/current/hello-world/sample-application.html

The sample application consists of 3 components:

- **Couchbase Server (we will be using 6.6.5)**
- **Backend API (we will be implementing)**
- **Frontend Application (premade)**

The backend API will be based on the C SDK - libcouchbase (https://github.com/couchbase/libcouchbase) and use Kore.io for the web server framework.

The sample application contains a lot of “wrapper code“ for configuration, error handling, communication which we are outside the scope of this workshop.

Documentation & Examples

Open the documentation for libcouchbase!

- https://docs.couchbase.com/c-sdk/current/hello-world/start-using-sdk.html
- https://docs.couchbase.com/c-sdk/current/howtos/kv-operations.html
- https://docs.couchbase.com/c-sdk/current/howtos/subdocument-operations.html
- https://docs.couchbase.com/c-sdk/current/howtos/n1ql-queries-with-sdk.html

- Open the try-cb-lcb project also => This is the solution.
- Open the try-cb-lcb-labs project => This is where you start.

Load and build the Project

For simplicity, we will be running all 3 Sample Application components as Docker containers. As we implement the API, we will rebuild the backend container with updated functionality.

To begin, if you have not already done so, clone the workshop git repo:

- git clone https://github.com/couchbase-ps/cb-workshop-2d

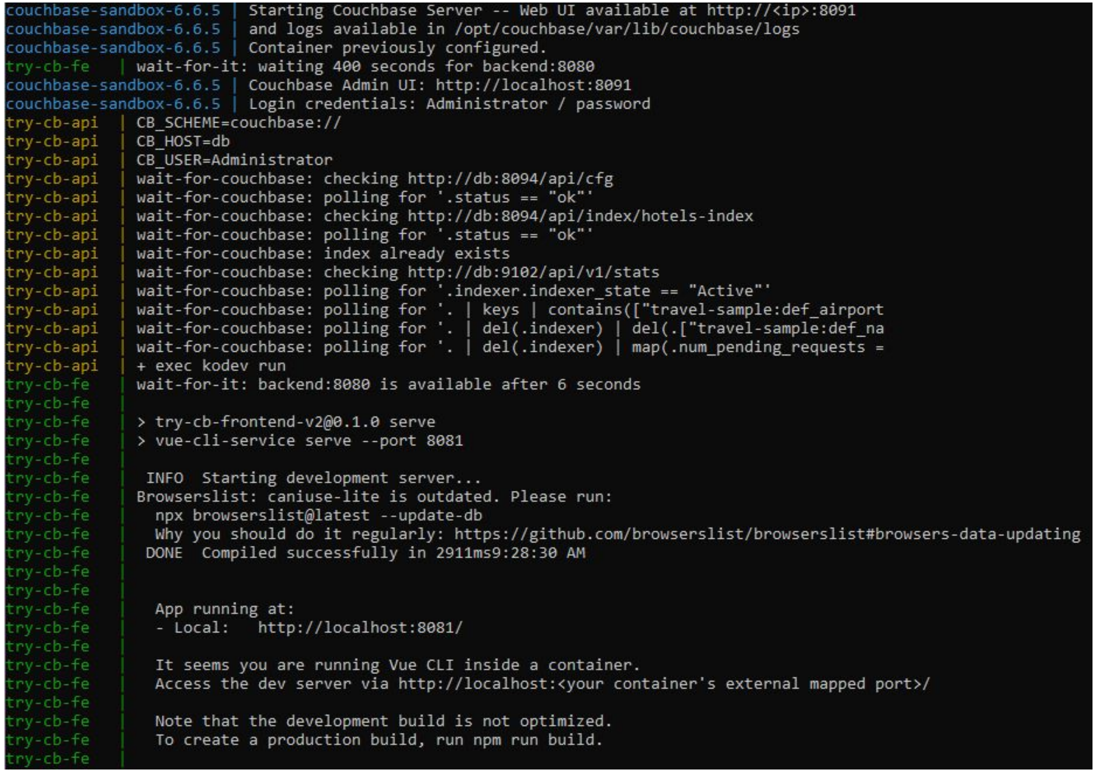
Open your preferred IDE and load the folder `` `*cb-workshop-2d/cpp/try-cb-lcb-labs*` `` as your working directory.

You may additionally load `` `*cb-workshop-2d/cpp/try-cb-lcb*` `` as a reference.

Inside the working directory, run `` `docker-compose up --build` `` to initialize the Docker instances. You will note in the docker-compose.yml file that the backend API is build locally while the other components are remote.

Verify Build Process

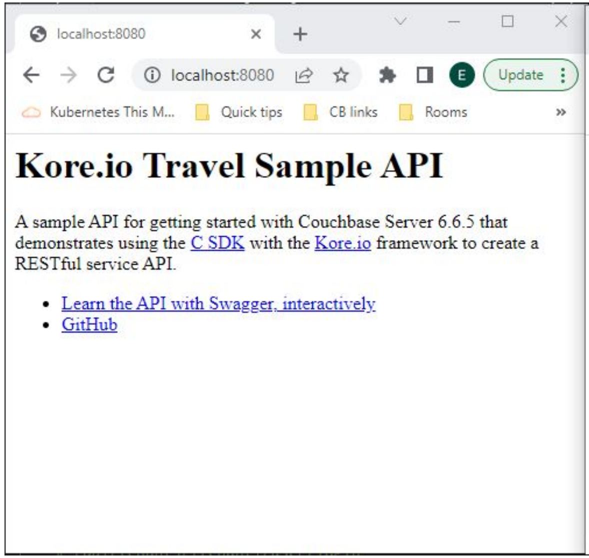
If all components loaded correctly, your terminal should show the following:



```
couchbase-sandbox-6.6.5 | Starting Couchbase Server -- Web UI available at http://<ip>:8091
couchbase-sandbox-6.6.5 | and logs available in /opt/couchbase/var/lib/couchbase/logs
couchbase-sandbox-6.6.5 | Container previously configured.
try-cb-fe    | wait-for-it: waiting 400 seconds for backend:8080
couchbase-sandbox-6.6.5 | Couchbase Admin UI: http://localhost:8091
couchbase-sandbox-6.6.5 | Login credentials: Administrator / password
try-cb-api   | CB_SCHEME=couchbase://
try-cb-api   | CB_HOST=db
try-cb-api   | CB_USER=Administrator
try-cb-api   | wait-for-couchbase: checking http://db:8094/api/cfg
try-cb-api   | wait-for-couchbase: polling for '.status == "ok"'
try-cb-api   | wait-for-couchbase: checking http://db:8094/api/index/hotels-index
try-cb-api   | wait-for-couchbase: polling for '.status == "ok"'
try-cb-api   | wait-for-couchbase: index already exists
try-cb-api   | wait-for-couchbase: checking http://db:9102/api/v1/stats
try-cb-api   | wait-for-couchbase: polling for '.indexer.indexer_state == "Active"'
try-cb-api   | wait-for-couchbase: polling for '. | keys | contains(["travel-sample:def_airport
try-cb-api   | wait-for-couchbase: polling for '. | del(.indexer) | del(.["travel-sample:def_na
try-cb-api   | wait-for-couchbase: polling for '. | del(.indexer) | map(.num_pending_requests =
try-cb-api   | + exec kodev run
try-cb-fe    | wait-for-it: backend:8080 is available after 6 seconds
try-cb-fe    |
try-cb-fe    | > try-cb-frontend-v2@0.1.0 serve
try-cb-fe    | > vue-cli-service serve --port 8081
try-cb-fe    |
try-cb-fe    |  INFO  Starting development server...
try-cb-fe    | Browserslist: caniuse-lite is outdated. Please run:
try-cb-fe    |   npx browserslist@latest --update-db
try-cb-fe    |   Why you should do it regularly: https://github.com/browserslist/browserslist#browsers-data-updating
try-cb-fe    |  DONE  Compiled successfully in 2911ms9:28:30 AM
try-cb-fe    |
try-cb-fe    |
try-cb-fe    |   App running at:
try-cb-fe    |   - Local:   http://localhost:8081/
try-cb-fe    |
try-cb-fe    |   It seems you are running Vue CLI inside a container.
try-cb-fe    |   Access the dev server via http://localhost:<your container's external mapped port>/
try-cb-fe    |
try-cb-fe    |   Note that the development build is not optimized.
try-cb-fe    |   To create a production build, run npm run build.
try-cb-fe    |
```

Verify Build Process

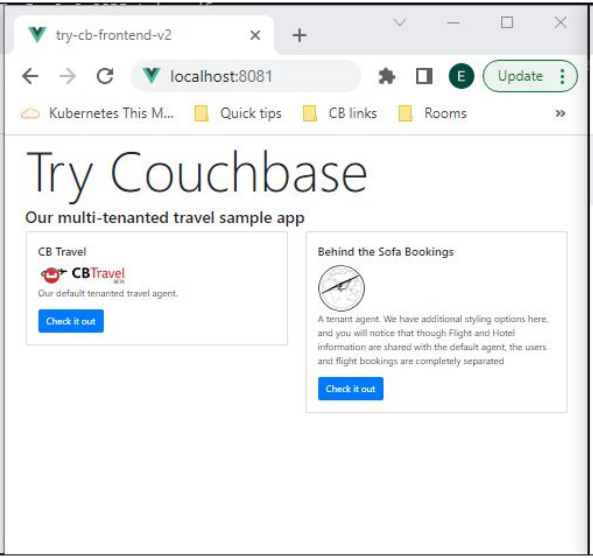
You should be able to access the the difference components on your localhost.
8080 - API, 8081 - web app, 8091 - Couchbase



Kore.io Travel Sample API

A sample API for getting started with Couchbase Server 6.6.5 that demonstrates using the C SDK with the Kore.io framework to create a RESTful service API.

- Learn the API with Swagger, interactively
- GitHub



Try Couchbase

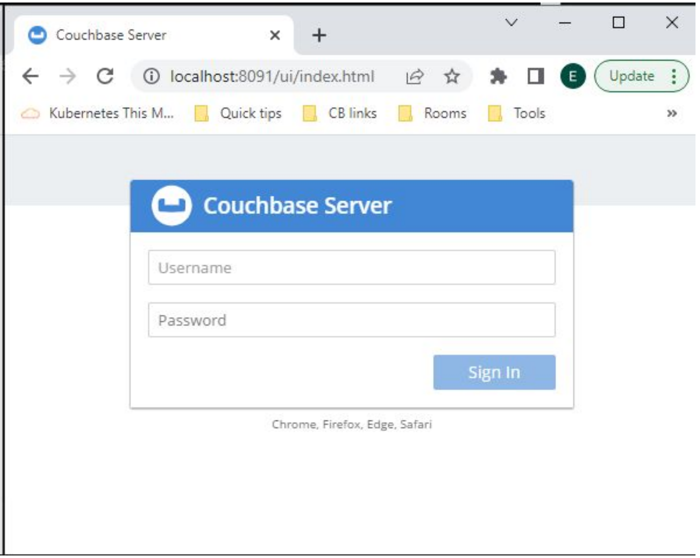
Our multi-tenanted travel sample app

CB Travel

Our default tenanted travel agent.

Behind the Sofa Bookings

A tenant agent. We have additional styling options here, and you will notice that though Flight and Hotel information are shared with the default agent, the users and flight bookings are completely separated

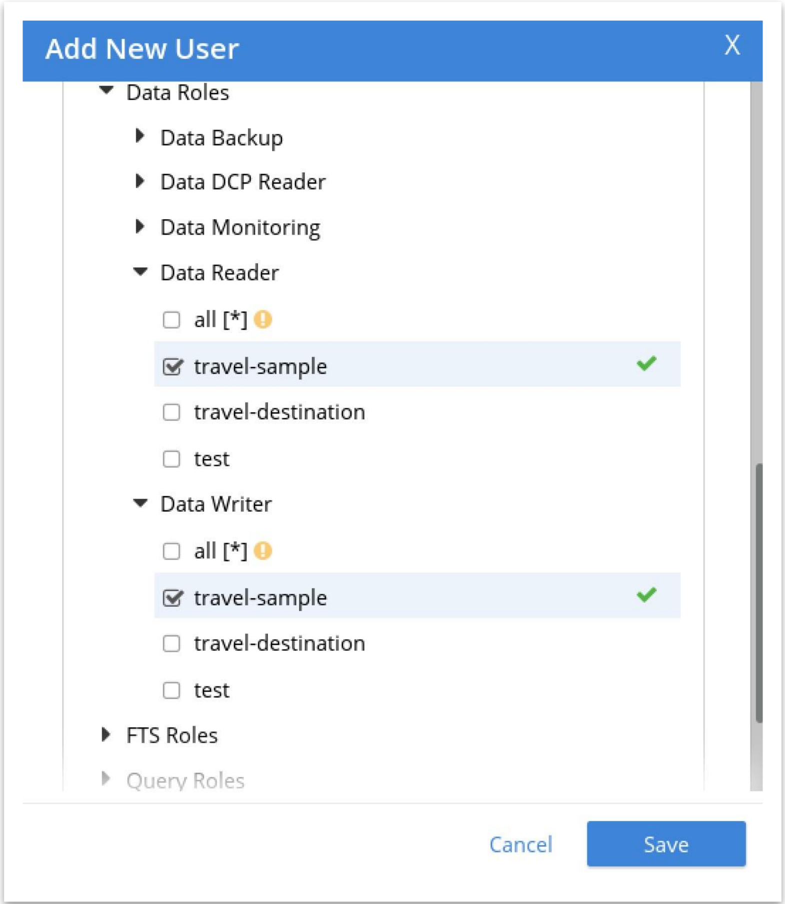


2 Managing Connections

Create an applicative User via RBAC

Perform the following operations:

- Browse to the Couchbase Server UI http://localhost:8091/
- Go the Security Tab in Couchbase
- Create a new User.
- Username = “application”
- Password = “password”
- Roles
  - Data Reader on `travel-sample`
  - Data Writer on `travel-sample`
  - Query Select on `travel-sample`



Add New User

- Data Roles
  - Data Backup
  - Data DCP Reader
  - Data Monitoring
  - Data Reader
    - ☐ all [*]
    - ☑ travel-sample ✓
    - ☐ travel-destination
    - ☐ test
  - Data Writer
    - ☐ all [*]
    - ☑ travel-sample ✓
    - ☐ travel-destination
    - ☐ test
- FTS Roles
- Query Roles

Connecting to Couchbase with RBAC

In the try-cb-lcb.c source code file, edit the kore_worker_configure() method:
Look for the "// LAB – Couchbase bootstrap" comments to implement the following:

- Configure the connection
- Create the instance
- Connect to the cluster
- Check the bootstrap status
- Install callbacks
- Open bucket

Edit the destroy_cb_instance() method to cleanup and destroy the couchbase connection instance:
Look for the "// LAB – Couchbase shutdown" comment to implement

- Destroy connection/instance

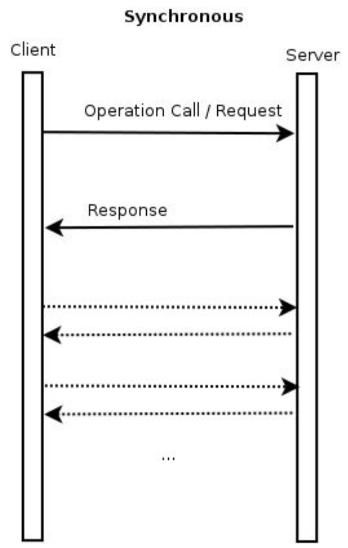
Rebuild the project

Test your implementation by calling the Test API: localhost:8080/api/test
This will perform a basic k/v GET from the `travel-sample` bucket.

3 Working with Documents

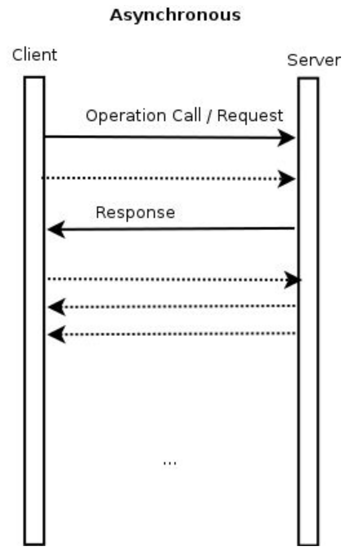
SDK APU Blocking

- Libcouchbase is designed to use non-blocking I/O
  - Scheduled operations
- But lcb_wait() blocks by default
  - Waits for pending requests
  - Used for synchronous operation execution
- Callback functions are used
  - e.g. storage_callback



SDK API Non-Blocking

- External event loop integration
  - Provides mechanism to execute a callback function when a specific event occurs
- Asynchronous operation execution
- No need for lcb_wait()



Insert a Document

Make sure that the travel-sample data is installed!
Edit the api-user-auth.c source file:
Modify the insert_user() method to insert a new document:
Look for the "// LAB – Insert" comments to implement the following:

- Create the command
- Set the document ID
- Set the document value
- Store the document

If running against a Couchbase 7+ cluster, add functionality to specify the *collection/scope*.

Rebuild the project

Open your browser to the travel sample app website: http://localhost:8081/.

Choose the CBTravel application and register a new user account:

username: travel, password: 123456

Verify Insert Operation

Browse to the Couchbase Server UI at http://localhost:8091/
Enter
Look for the "// LAB – Insert" comments to implement the following:

- Create the command
- Set the document ID
- Set the document value
- Store the document

If running against a Couchbase 7+ cluster, add functionality to specify the *collection/scope*.

Rebuild the project

Open your browser to the travel sample app website: http://localhost:8081/

Choose the CBTravel application and register a new user account:

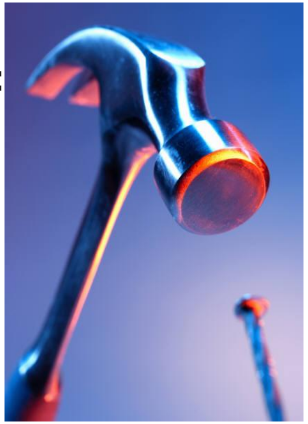
username: travel, password: 123456

Get a (Sub)Document

Edit the get_user_password() method:

Look for the "// LAB – Get Subdoc" comments to implement the following:

- Create the Command
- Specify the Document ID
- Create Subdoc Spec
- Specify the Subdoc Field to return
- Add Subdoc operations to command
- Run Subdoc operation



If running against a Couchbase 7+ cluster, add functionality to specify the *collection/scope*.

Rebuild the project

Open your browser to the travel sample app website: http://localhost:8081/ and choose the CBTravel application.

In the front-end application, login using the user account you added in the previous step.

Create/Update a Document

Edit the api-user-flights.c source file:
Modify the upsert-new-flight() method to upsert a new document:
Look for the "// LAB – Upsert" comments to implement the following:

- Create the command
- Set the document ID
- Set the document value
- Store the document

Notice the similarity to Inserting a new document!

Unfortunately, this and the subsequent steps will not be testable until the N1QL query step has been completed.

Update a (Sub)Document

Modify the add_user_booking() method:

Look for the "// LAB – Update Subdoc" comments to implement the following:

- Create Command
- Specify the Document ID
- Create Subdoc Spec
- Specify the Subdoc Field to append to the array
- Add Subdoc operations to command
- Schedule Subdoc operation

Get a (Sub)Document

Modify the get_user_bookings() method:

Look for the "// LAB – Get Subdoc" comments to implement the following:

- Create Command
- Specify the Document ID
- Create Subdoc Spec
- Specify the Subdoc Field to return
- Add Subdoc operations to command
- Schedule Subdoc operation

Get a Document

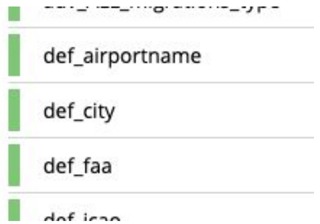
Modify the get_flight_booking() method:

Look for the “// LAB – Get Document” comments to implement the following:

- Create Command
- Specify the Document ID
- Perform Get

4 | N1QL Queries

Query via SQL++


def_airportname

def_city

def_faa

Make sure that the Secondary Indexes on 'faa' and 'airportname' are there!

Edit the api-flight-paths.c source file:

Modify the tcblcb-api-fpaths() method to query using positional parameters:

Look for the "// LAB – Position Param Query" comments to implement the following:

- Create the Query command
- Add the SQL++ query to the query command
- Add the positional parameters to the query
- Set query formatting
- Specify Adhoc = FALSE
- Set the callback function
- Send the query to the cluster

Query via SQL++

Continue modifing the tcblcb-api-fpaths() method to query using positional parameters:
Look for the "// LAB – Named Param Query" comments to implement the following:

- Reset the query command
- Create the Query command
- Add the SQL++ to the query command
- Add named parameters to the query
- Set query formatting
- Specify Adhoc = FALSE
- Set the callback function
- Send the query to the cluster

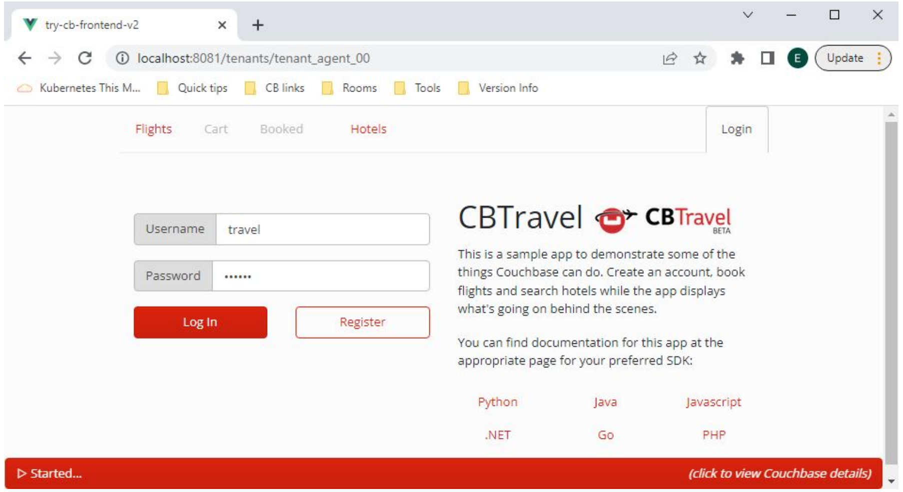
5 Travel Sample Application

A Sample Application

Rebuild the project

Open your browser to the travel sample app website: http://localhost:8081/ choose the CBTravel application.

Log in using the ‘travel’ user you registered.



try-cb-frontend-v2

localhost:8081/tenants/tenant_agent_00

Kubernetes This M... Quick tips CB links Rooms Tools Version Info

Flights Cart Booked Hotels Login

Username: travel

Password: ••••••

Log In Register

CBTravel

This is a sample app to demonstrate some of the things Couchbase can do. Create an account, book flights and search hotels while the app displays what's going on behind the scenes.

You can find documentation for this app at the appropriate page for your preferred SDK:

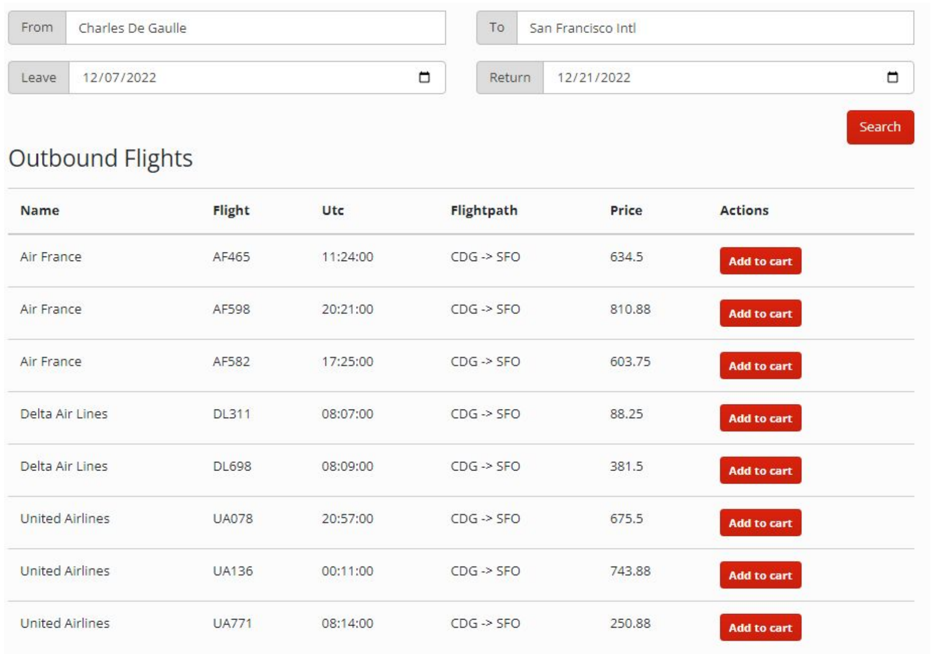
Python Java Javascript

.NET Go PHP

▷ Started... *(click to view Couchbase details)*

A Sample Application

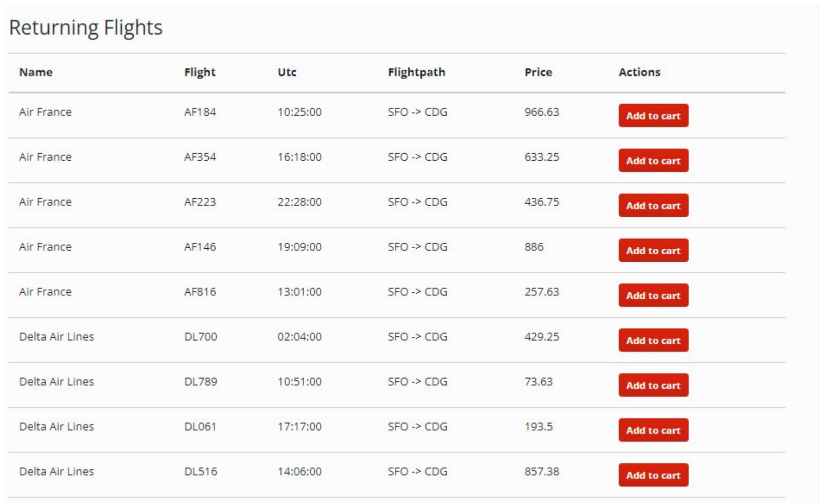
Search for flights from “Charles De Gaulle” to “San Francisco Intl” (or other airports of your choice, using the airport name). Specify travel dates. Select one or more flights from both the outbound and returning flights and add to cart.



From	Charles De Gaulle
To	San Francisco Intl
Leave	12/07/2022
Return	12/21/2022

Outbound Flights

Name	Flight	Utc	Flightpath	Price	Actions
Air France	AF465	11:24:00	CDG -> SFO	634.5	Add to cart
Air France	AF598	20:21:00	CDG -> SFO	810.88	Add to cart
Air France	AF582	17:25:00	CDG -> SFO	603.75	Add to cart
Delta Air Lines	DL311	08:07:00	CDG -> SFO	88.25	Add to cart
Delta Air Lines	DL698	08:09:00	CDG -> SFO	381.5	Add to cart
United Airlines	UA078	20:57:00	CDG -> SFO	675.5	Add to cart
United Airlines	UA136	00:11:00	CDG -> SFO	743.88	Add to cart
United Airlines	UA771	08:14:00	CDG -> SFO	250.88	Add to cart

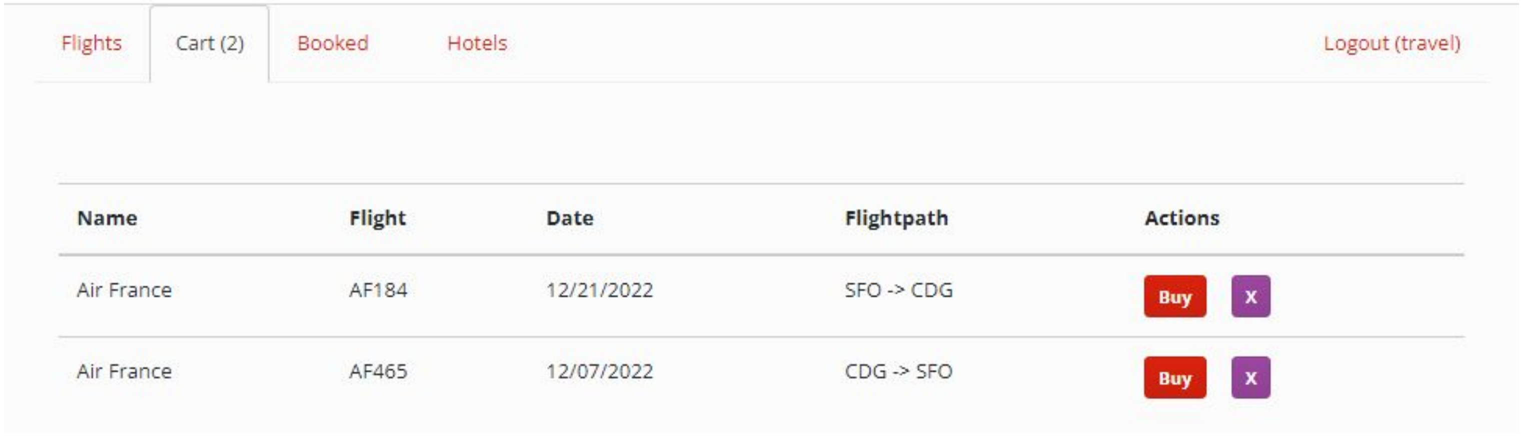


Returning Flights

Name	Flight	Utc	Flightpath	Price	Actions
Air France	AF184	10:25:00	SFO -> CDG	966.63	Add to cart
Air France	AF354	16:18:00	SFO -> CDG	633.25	Add to cart
Air France	AF223	22:28:00	SFO -> CDG	436.75	Add to cart
Air France	AF146	19:09:00	SFO -> CDG	886	Add to cart
Air France	AF816	13:01:00	SFO -> CDG	257.63	Add to cart
Delta Air Lines	DL700	02:04:00	SFO -> CDG	429.25	Add to cart
Delta Air Lines	DL789	10:51:00	SFO -> CDG	73.63	Add to cart
Delta Air Lines	DL061	17:17:00	SFO -> CDG	193.5	Add to cart
Delta Air Lines	DL516	14:06:00	SFO -> CDG	857.38	Add to cart

A Sample Application

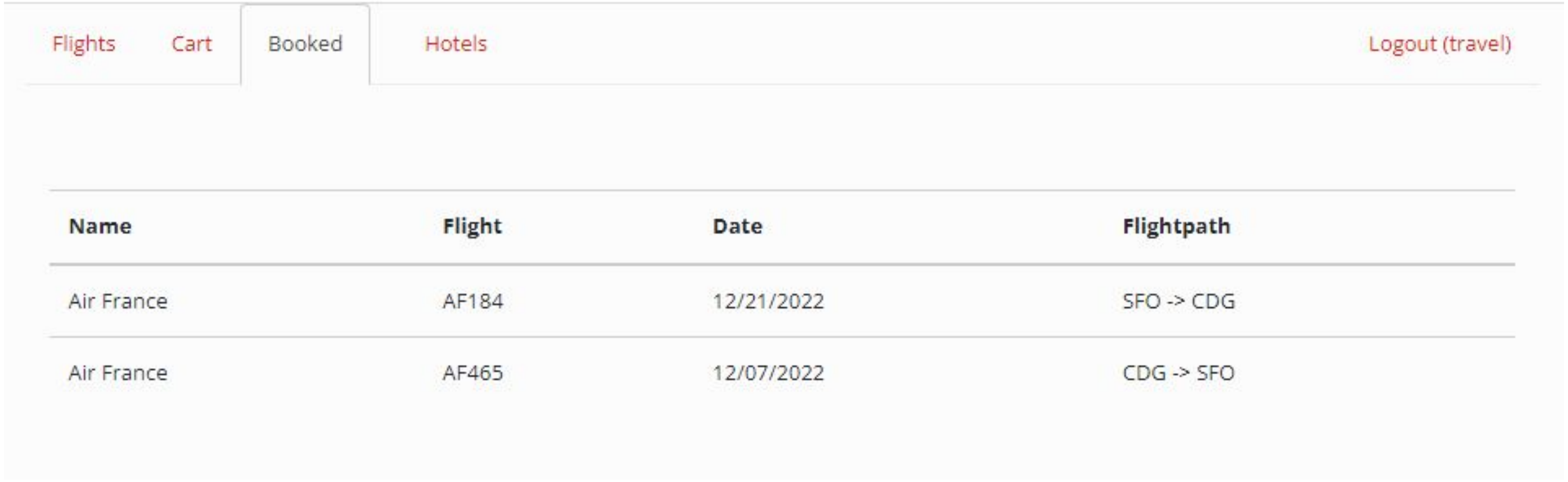
In your cart, click to buy one or more flights.



Name	Flight	Date	Flightpath	Actions
Air France	AF184	12/21/2022	SFO -> CDG	Buy X
Air France	AF465	12/07/2022	CDG -> SFO	Buy X

A Sample Application

The purchased flights should appear in your booked flights page.



Name	Flight	Date	Flightpath
Air France	AF184	12/21/2022	SFO -> CDG
Air France	AF465	12/07/2022	CDG -> SFO

Thank you

Couchbase