Snap Script Implementation Details

Team Selector

Mentor: Yogesh Gangwar
Intern : Arya Wankhede
Buddy: Pragati Yadav

Project Introduction

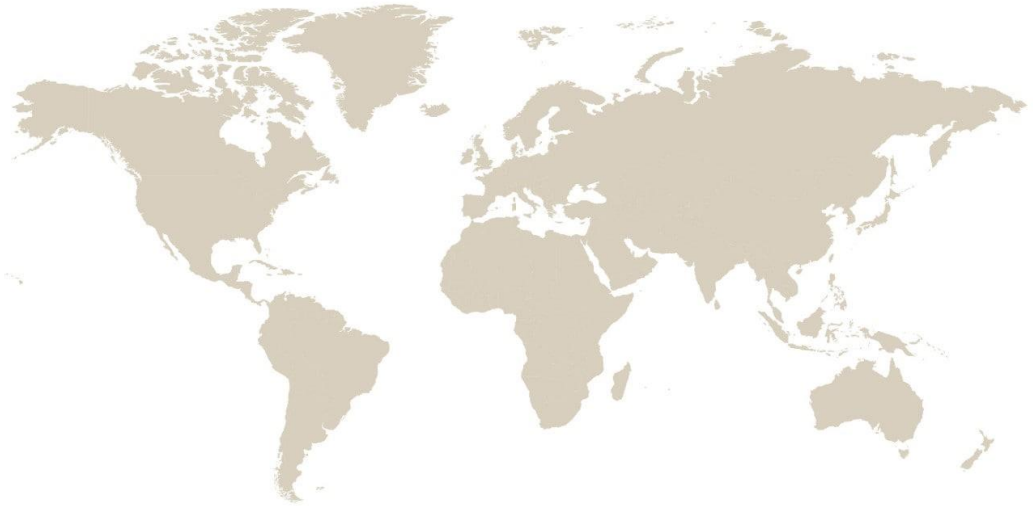
Snap-Script – A utility for UL Countries (US and Canada) to automate missing product system images script generation.
Today, missing product system images require manual database scripting, which is time-consuming and error-prone.
Snap-Script provides developers a quick glance at missing images through an intuitive UI and allows them to generate the necessary database scripts with a single click — saving time and boosting productivity.

Goal/Deliverables :

- Allow developers to view all missing product system images for UL countries with a single click
- Automate the generation of database scripts for missing images through a single-click action

Business Impact

The solution will be integrated into the Firestop Selector’s in-house tools ecosystem, optimizing internal workflows.
By shortening development time, it will directly contribute to accelerating the time-to-market for related products and updates.

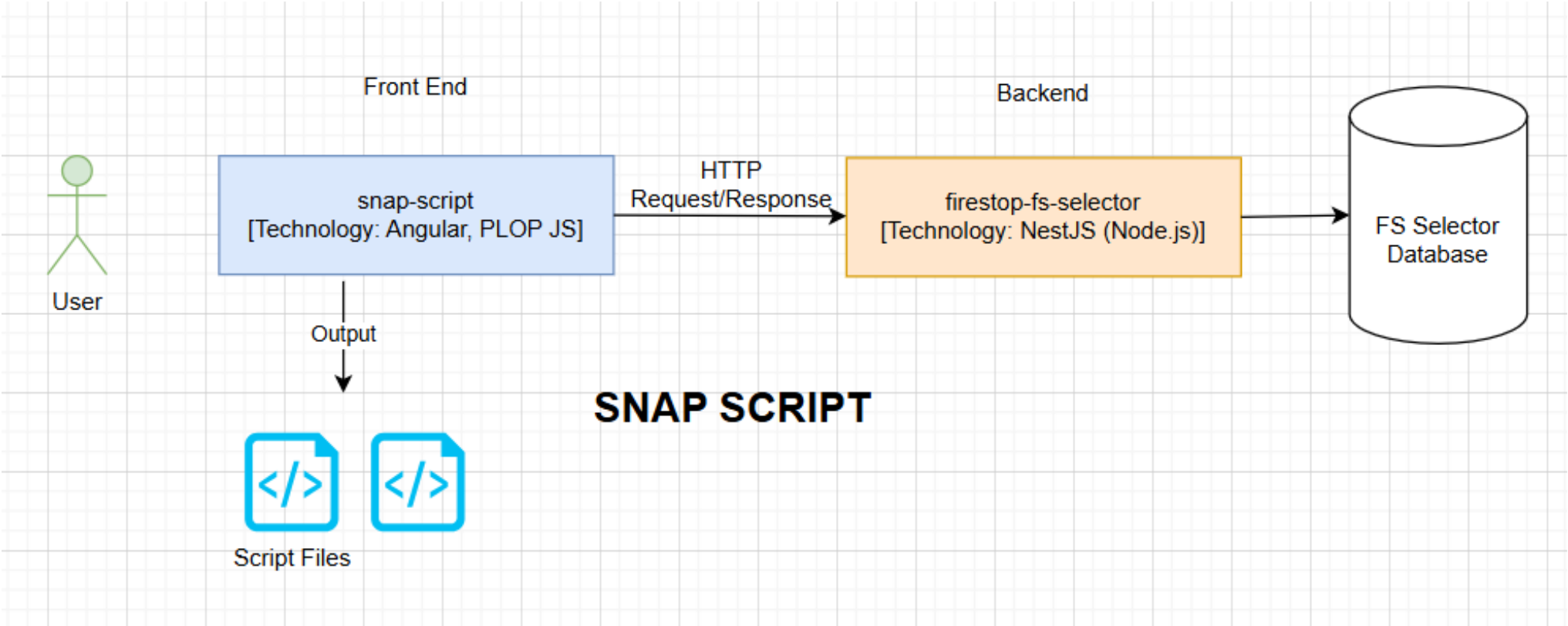


Pre-requisite:

Technology Stack:
Angular, Nest js, Plop, VS Code IDE , Jest Framework, Jira , GitLab, Postgres database, Docker

Existing ETL Process
https://hilti.atlassian.net/wiki/spaces/BFS/pages/1704100926/ETL+Run+-+Quick+Steps

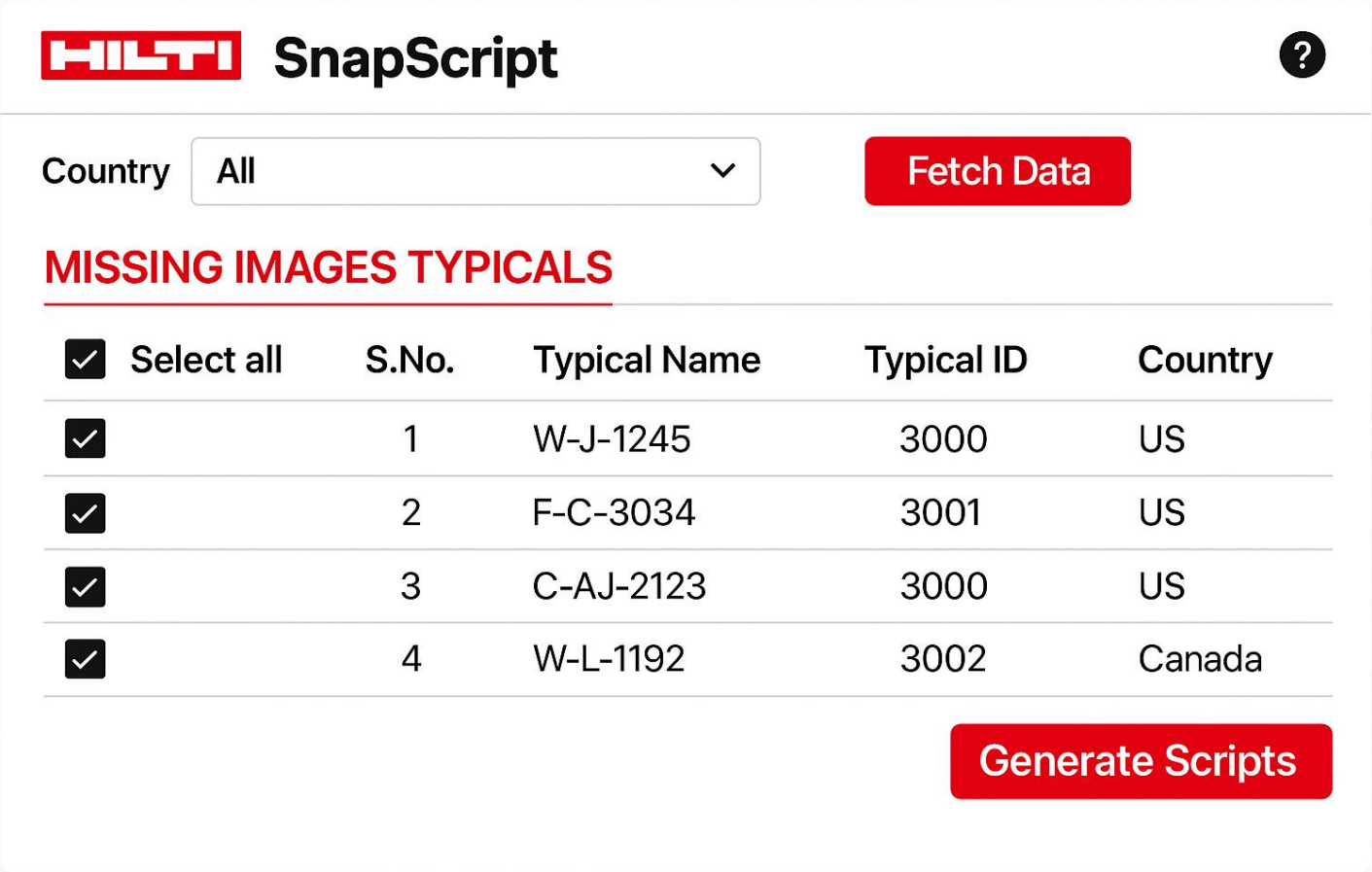
Implementation Approach



Implementation Approach Backend

Frontend UI:

Create new standalone **snap-script** angular project. And create components to provide UX as shown in below mock-up.



Implementation Approach Backend

Frontend UI Functionality:

1. The UI **should have a dropdown** to select the country (US, Canada, or All) and a button to fetch data for the selected option.

2. It **should display document data** in a grid with the following columns: **S.No., Typical Name, Typical ID, Market ID, and Type**.

3. The user **should be able to select all documents or specific ones** based on their requirement.

4. After selection, the user **should be able to click on the "Generate Scripts" button** to generate the corresponding database script.

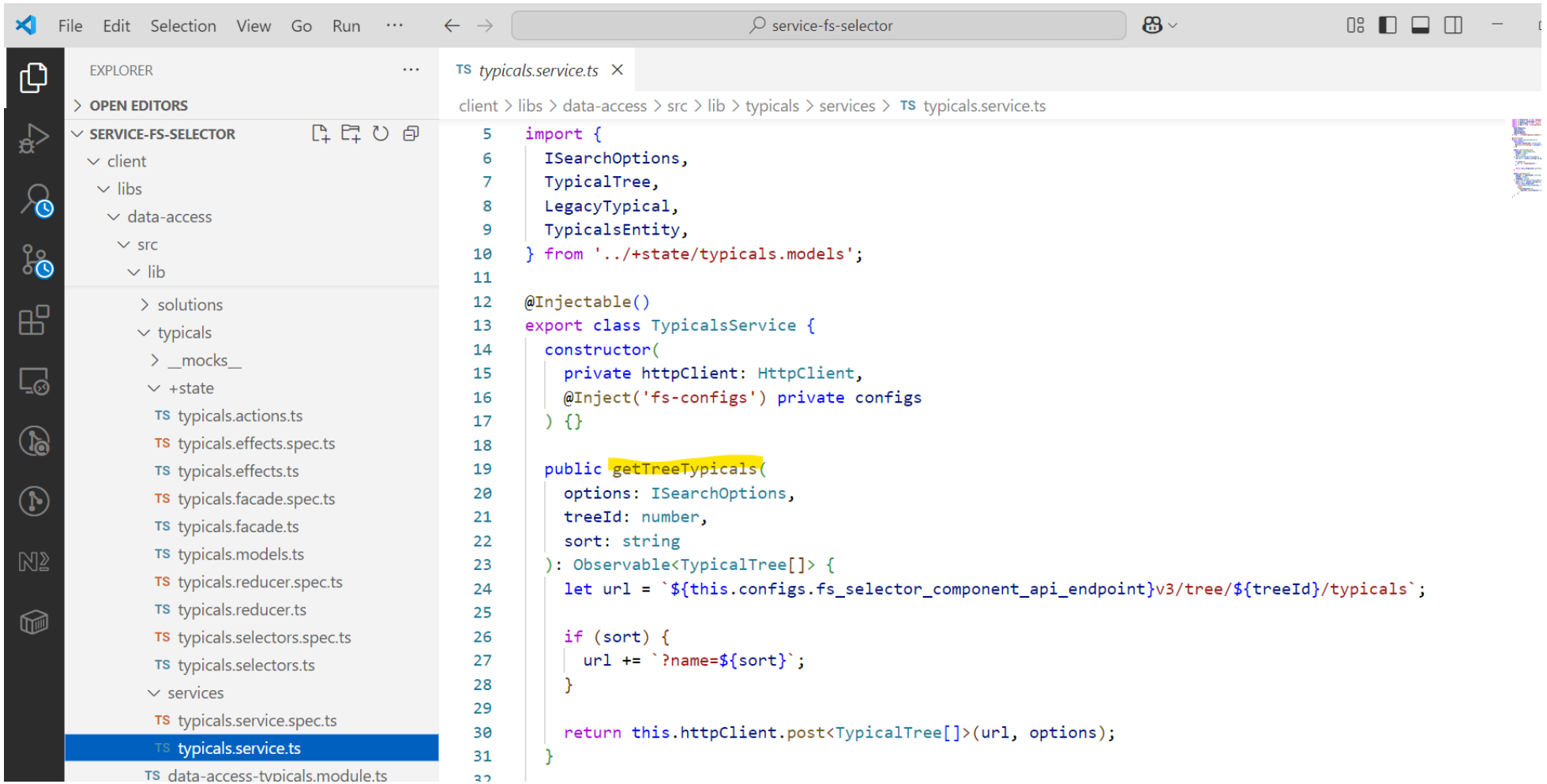
5. Data base script should generate in individual file for each selection.

6. Application should retain the state of Documents for which script generation successful.

Implementation Approach Backend

Frontend UI:

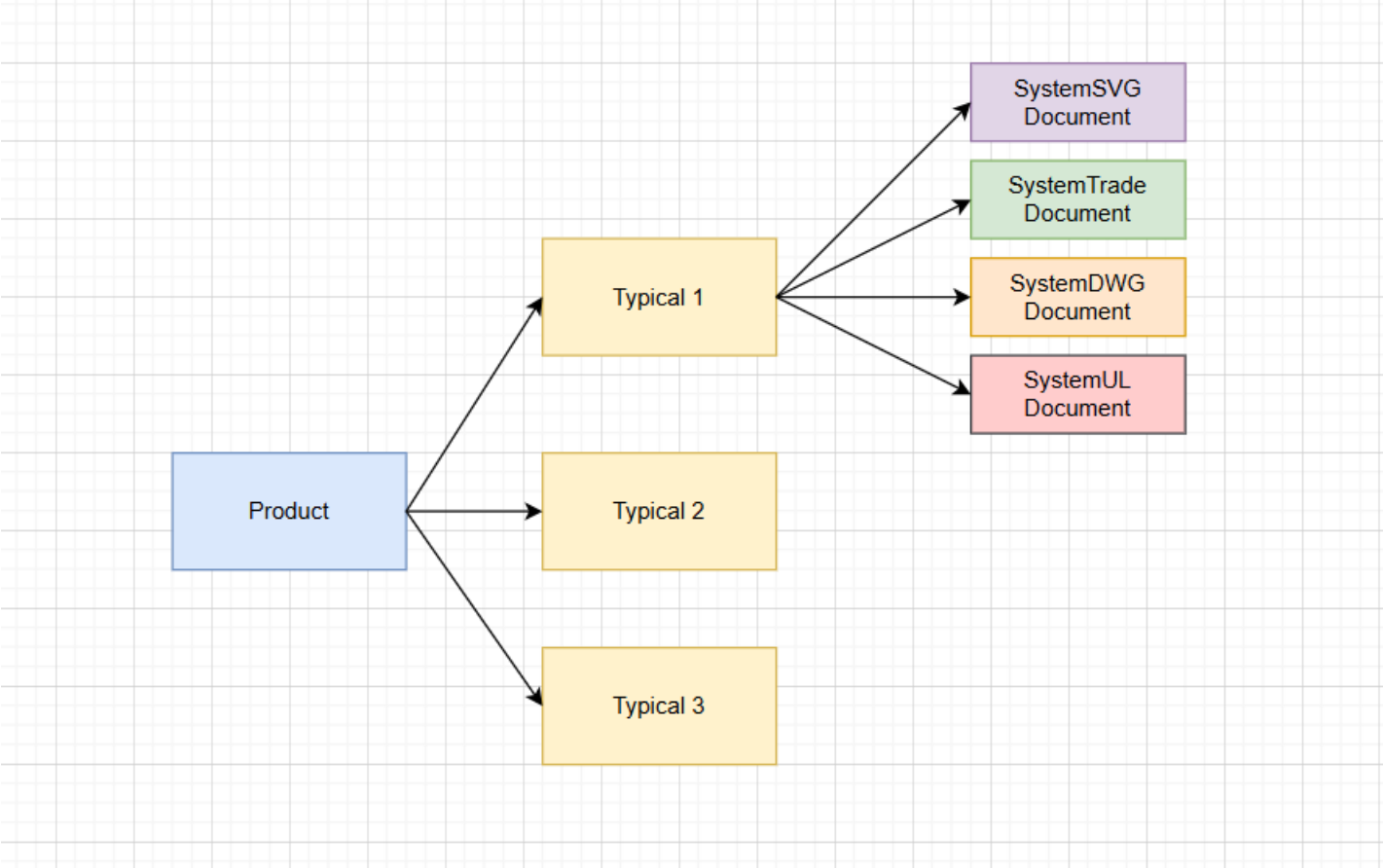
Create new service in **snap-script** angular project by taking reference of **TypicalsService** in **service-fs-selector.** Also use ngrx store to persist the data and execute actions on store.



Implementation Approach

Documents Linkage:

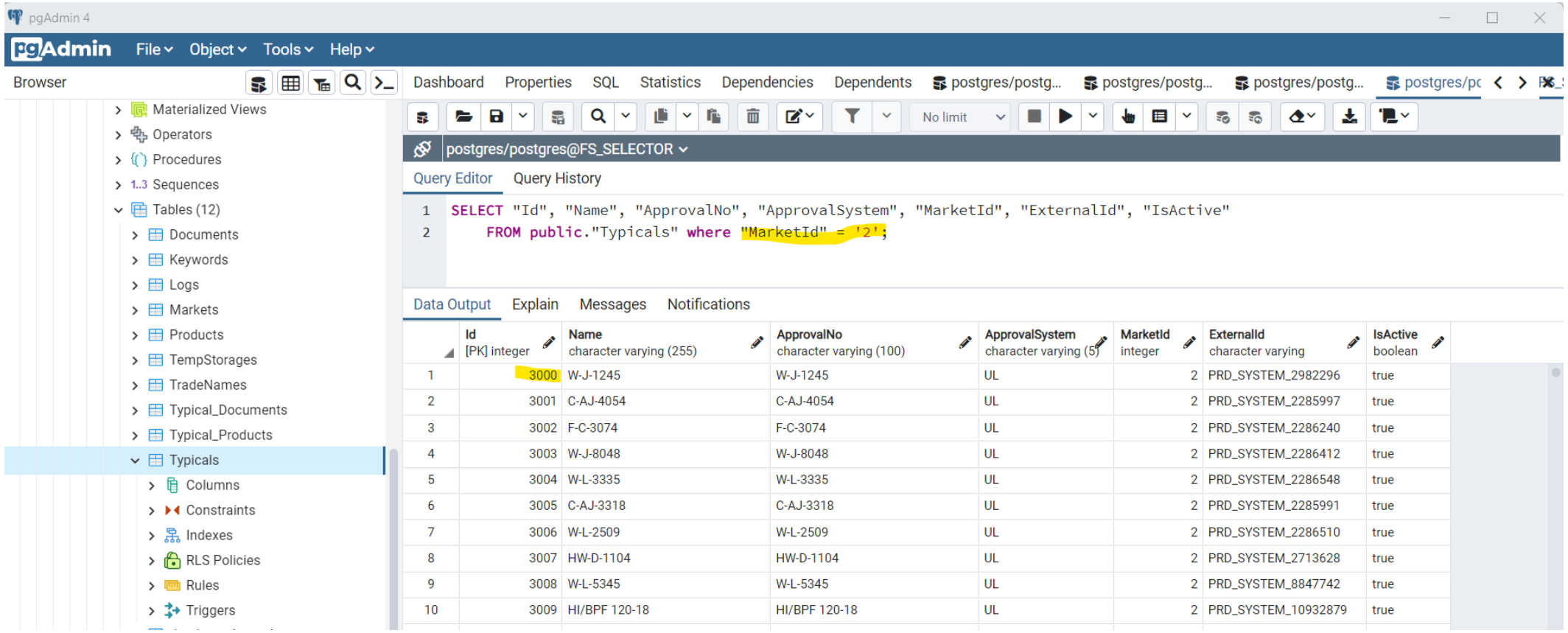
This is how multiple types Documents are linked to Typicals, which finally linked to Product



Implementation Approach Backend

Backend Database:

STEP 1] Open **Typicals** table in FS-Selector database in PostgreSQL. Get **Id's** from **Typicals** table for given **MarketId**



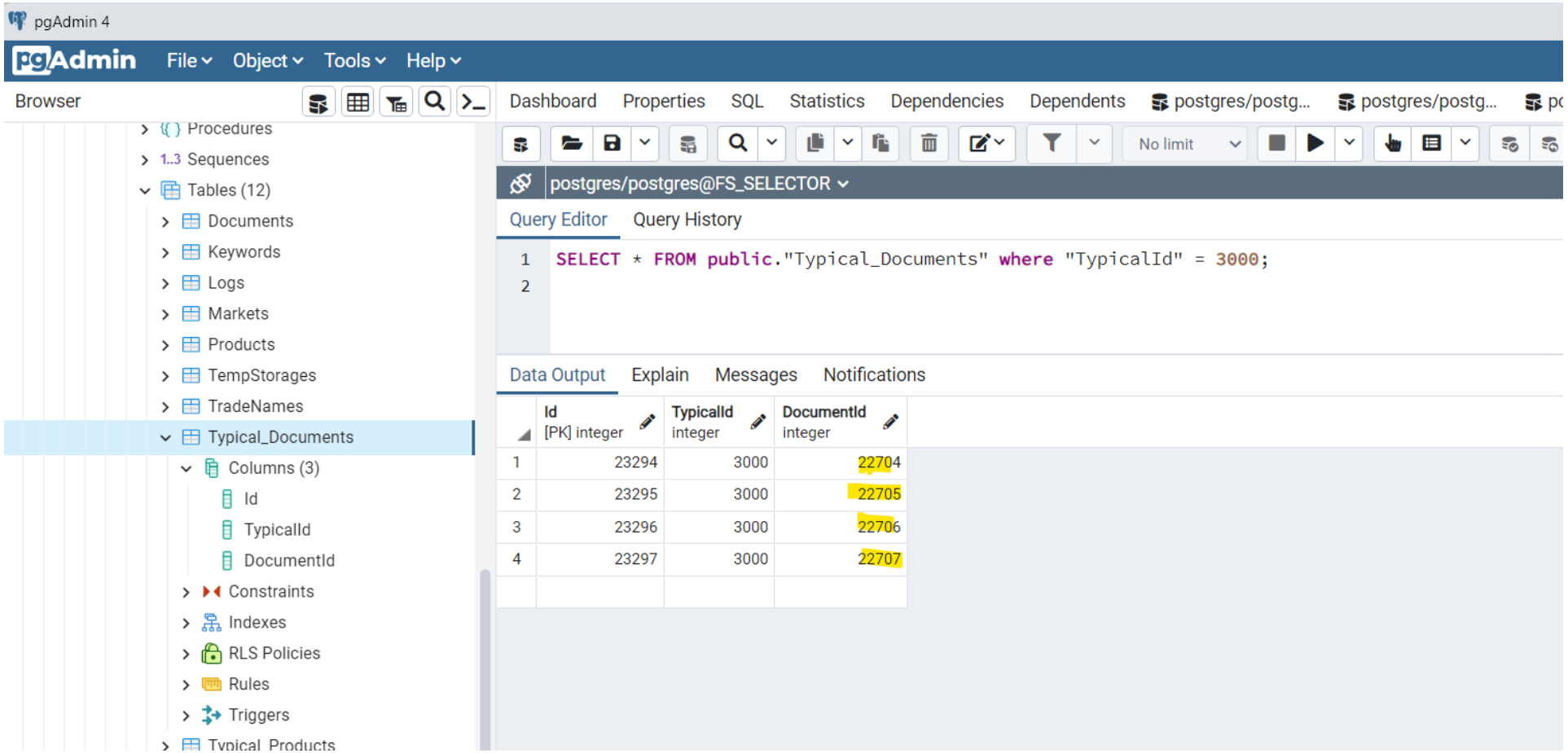
	Id [PK] integer	Name character varying (255)	ApprovalNo character varying (100)	ApprovalSystem character varying (5)	MarketId integer	ExternalId character varying	IsActive boolean
1	3000	W-J-1245	W-J-1245	UL	2	PRD_SYSTEM_2982296	true
2	3001	C-AJ-4054	C-AJ-4054	UL	2	PRD_SYSTEM_2285997	true
3	3002	F-C-3074	F-C-3074	UL	2	PRD_SYSTEM_2286240	true
4	3003	W-J-8048	W-J-8048	UL	2	PRD_SYSTEM_2286412	true
5	3004	W-L-3335	W-L-3335	UL	2	PRD_SYSTEM_2286548	true
6	3005	C-AJ-3318	C-AJ-3318	UL	2	PRD_SYSTEM_2285991	true
7	3006	W-L-2509	W-L-2509	UL	2	PRD_SYSTEM_2286510	true
8	3007	HW-D-1104	HW-D-1104	UL	2	PRD_SYSTEM_2713628	true
9	3008	W-L-5345	W-L-5345	UL	2	PRD_SYSTEM_8847742	true
10	3009	HI/BPF 120-18	HI/BPF 120-18	UL	2	PRD_SYSTEM_10932879	true

Implementation Approach Backend

Backend Database:

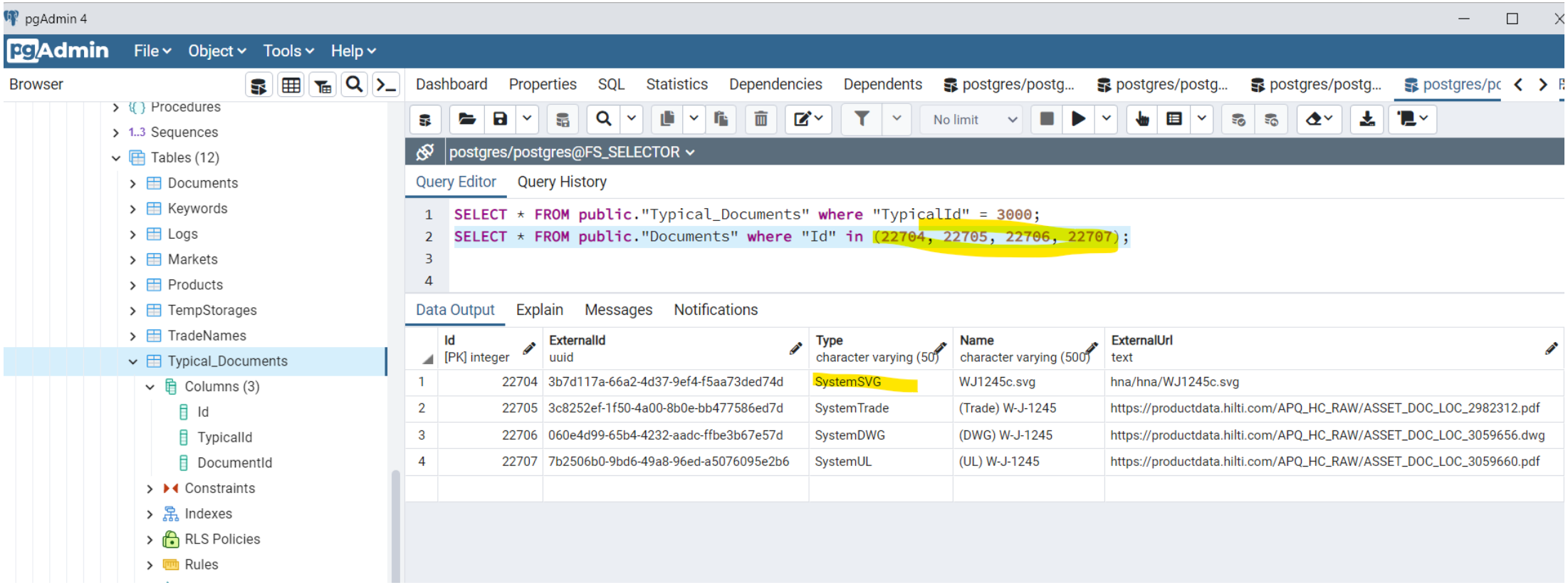
STEP 2] Open **Typical_Documents** table.

Get **DocumentId**'s from table for given **TypicalId**



Implementation Approach Backend

Backend Database:
STEP 3] Open **Documents** table.
Get **DocumentId**'s from table for given **TypicalId**



	Id [PK] integer	ExternalId uuid	Type character varying (50)	Name character varying (500)	ExternalUrl text
1	22704	3b7d117a-66a2-4d37-9ef4-f5aa73ded74d	SystemSVG	WJ1245c.svg	hna/hna/WJ1245c.svg
2	22705	3c8252ef-1f50-4a00-8b0e-bb477586ed7d	SystemTrade	(Trade) W-J-1245	https://productdata.hilti.com/APQ_HC_RAW/ASSET_DOC_LOC_2982312.pdf
3	22706	060e4d99-65b4-4232-aadc-ffbe3b67e57d	SystemDWG	(DWG) W-J-1245	https://productdata.hilti.com/APQ_HC_RAW/ASSET_DOC_LOC_3059656.dwg
4	22707	7b2506b0-9bd6-49a8-96ed-a5076095e2b6	SystemUL	(UL) W-J-1245	https://productdata.hilti.com/APQ_HC_RAW/ASSET_DOC_LOC_3059660.pdf

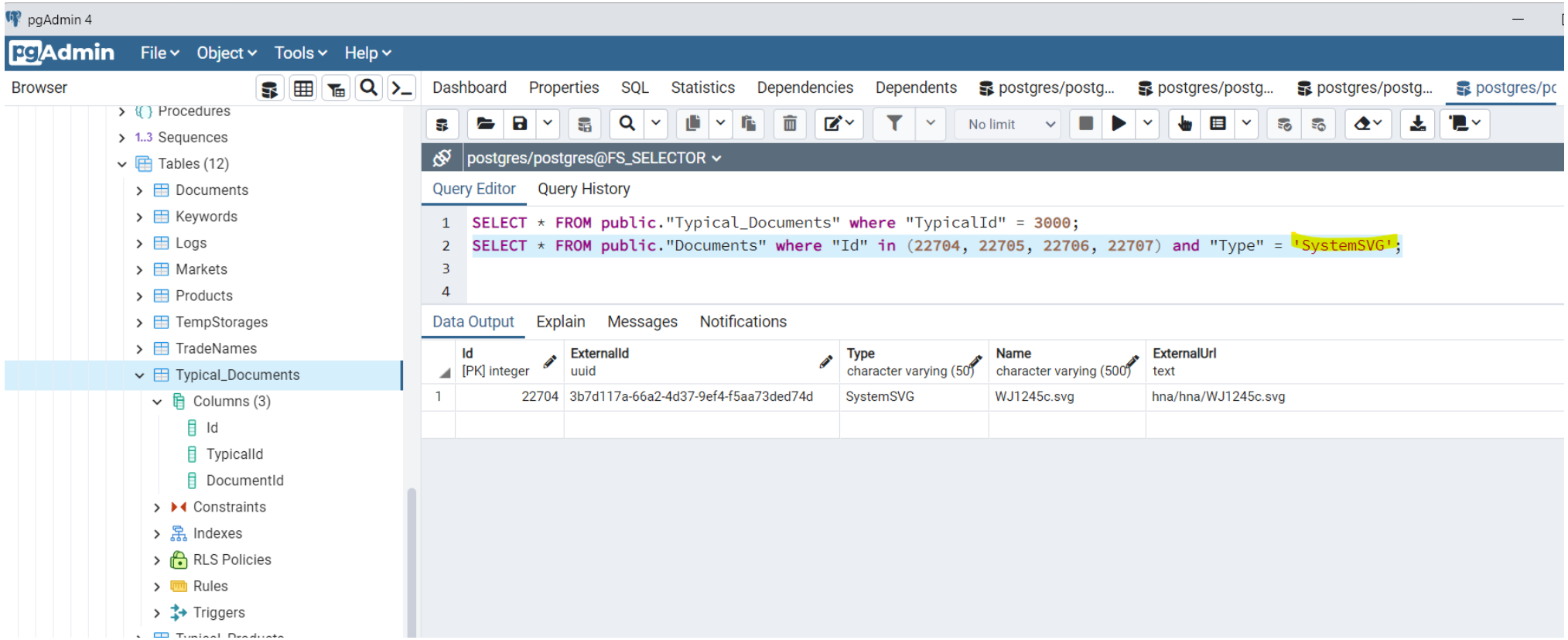
Implementation Approach Backend

Backend Database:

STEP 4] In **Documents** table.

For given **Id**' s collection, get record for which **Type** is '*SystemSVG*'

NOTE: If no records exist, it means we need to generate script for this.



Implementation Approach Backend

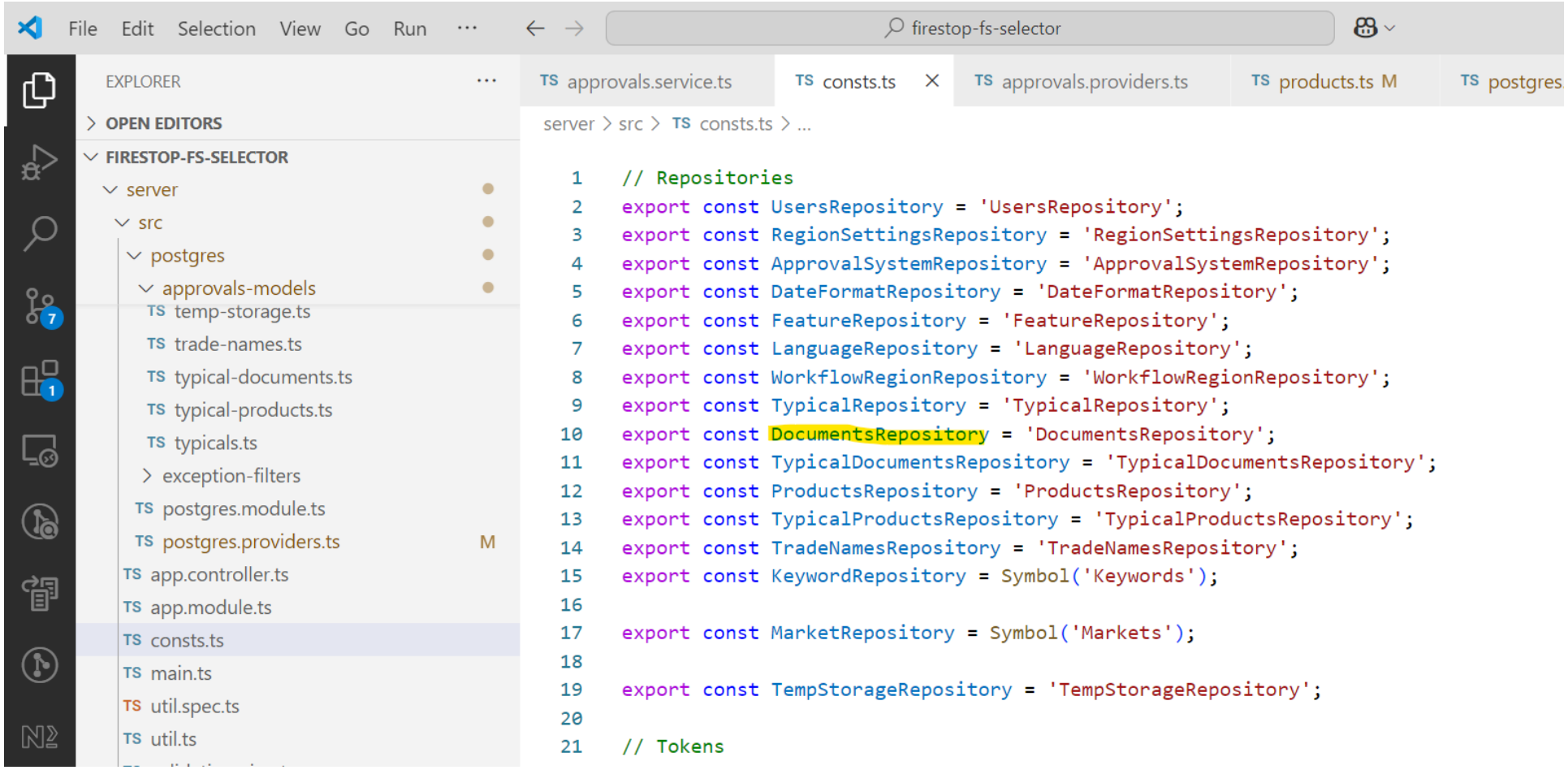
Backend Database:
Write query to fetch documents data where **ExternalUrl** is null or empty based on input **MarketId**
NOTE: For UL region, **Type** value should be ‘SystemSVG’

```
SELECT "Id", "Name", "ApprovalNo", "ApprovalSystem", "MarketId", "ExternalId",
"IsActive"    FROM public."Typicals" where "Id"= '4950’;

SELECT *   FROM public."Typical_Documents" where "TypicalId"= '4950’;

SELECT *   FROM public."Documents" where "Id" in (31741,31742, 31743, 31744,
34803);
```

Implementation Approach Backend

Backend Service:

Add service in **firestop-fs-selector project** using **DocumentsRepository**. Take reference of this repository and add new service **DocumentsService** and **DocumentsController.**



Implementation Approach Backend

Backend Service:
Add service in **firestop-fs-selector project** using **DocumentsRepository**. Take reference of this repository and add new service **DocumentsService** and **DocumentsController.**

Endpoint: `${this.configs.fs_selector_api_endpoint}documents/${treeId}/missingurl`;

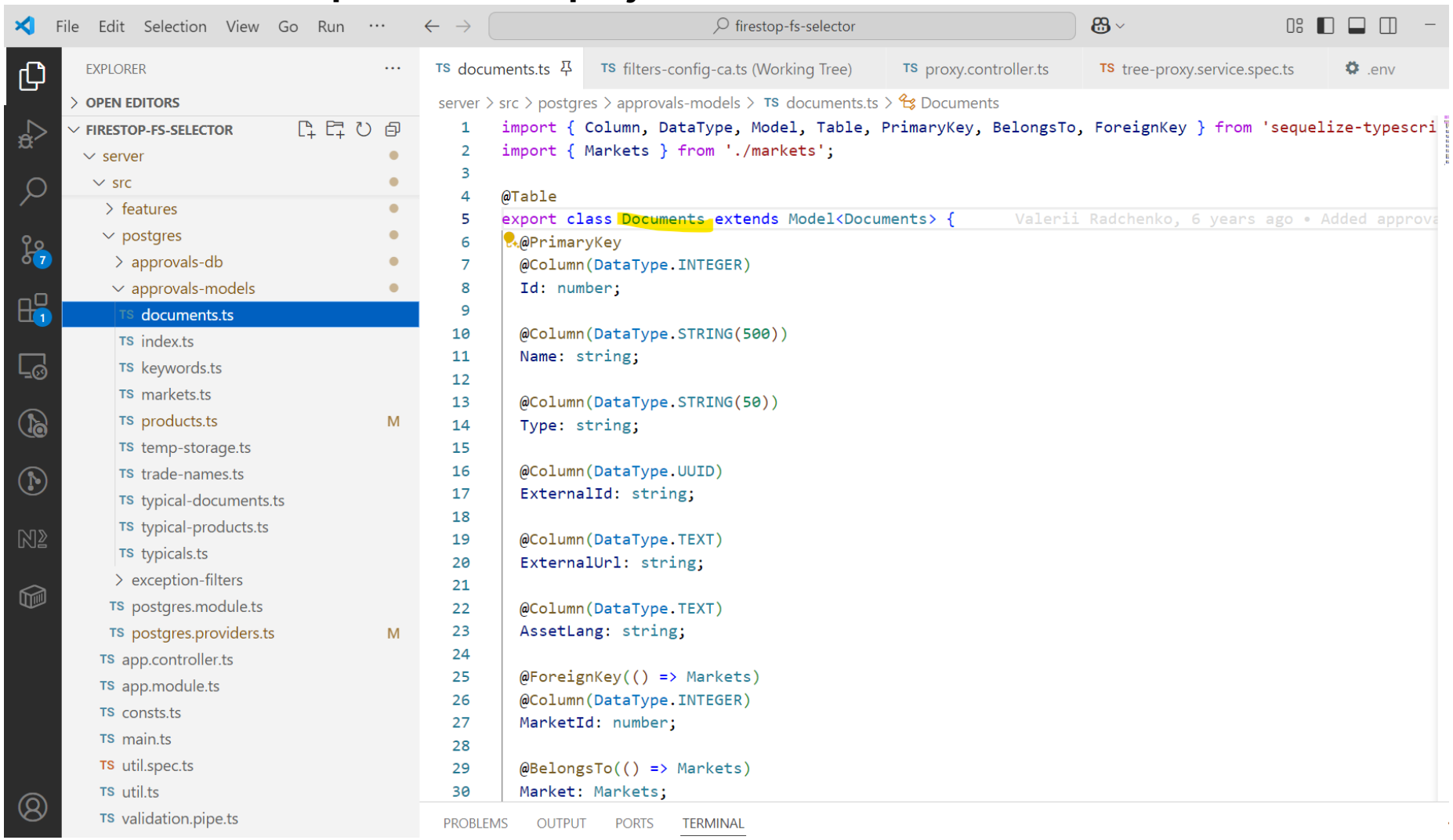
Documents/:treeId

```
@Controller('documents/:treeId')
@UseInterceptors(HcpTrackingInterceptor)
export class DocumentsController {
  constructor(private readonly documentsService: DocumentsService) {}

  @ApiBearerAuth()
  @Get('/missingurl')
  @UsePipes(ValidationPipe)
  @UseFilters(new ValidationExceptionFilter(), new ForeignKeyConstraintExceptionFilter())
  public async getMissingUrlDocuments(
    @Param('treeId', ConvertTreeIdToMarketIdPipe) marketId: number) {
    return this.documentsService.getMissingUrlDocuments(marketId);
  }
```

Implementation Approach Backend

Backend Service:

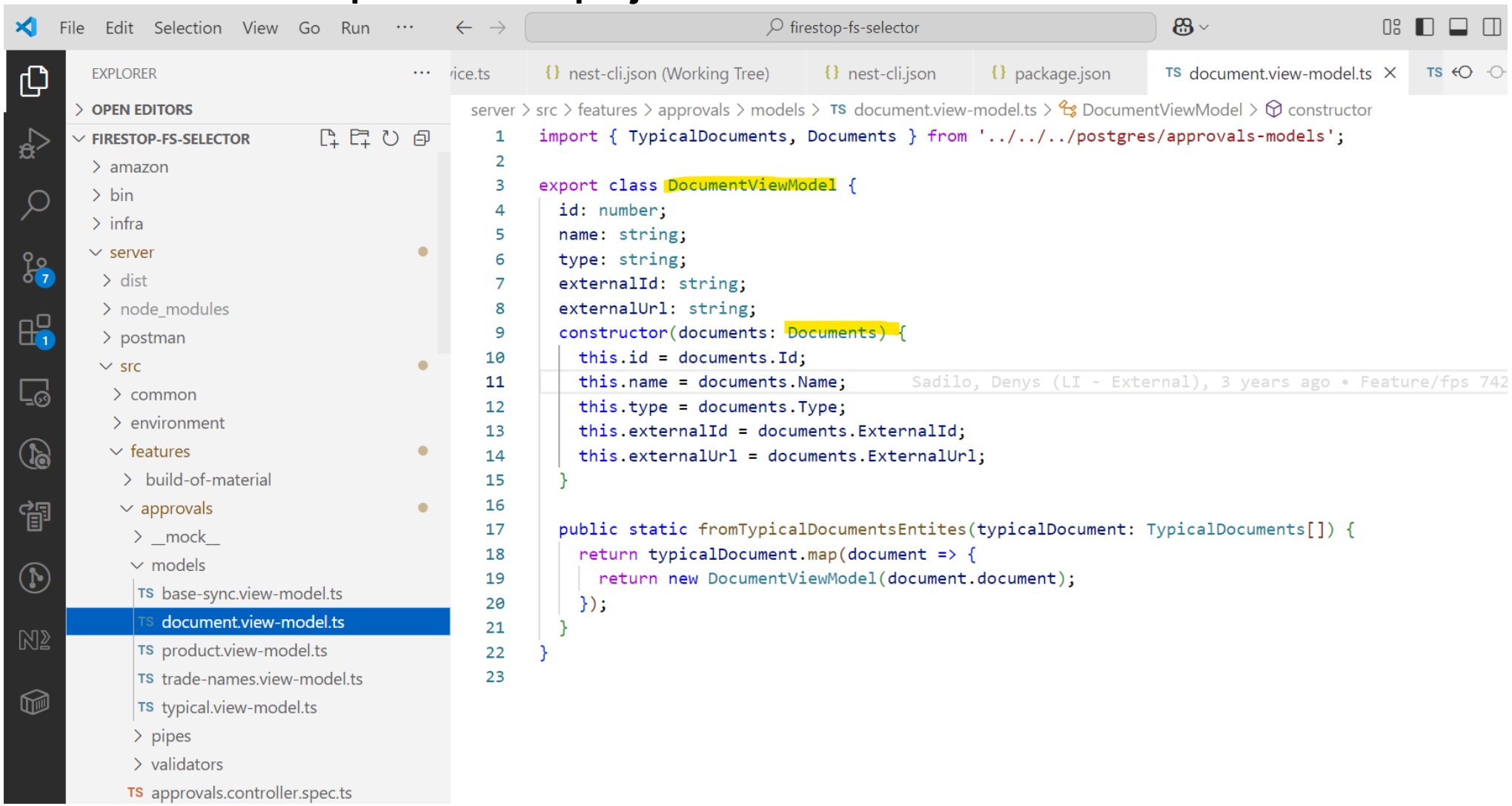
Add service in **firestop-fs-selector project**



Implementation Approach Backend

Backend Service:

Add service in **firestop-fs-selector project**



Implementation Approach Output Consumption

Backend Service:

Below sql script should be generated via Snap Script UI that will act as input for firestop-fs-selector

In Documents table:

Id should be **max id + 1**
ExternalId should be **guid**
Type should be **'SystemSVG'**
Name should be **Typical name**
ExternalUrl should be **Typical name** without **–** and additional **b.svg** suffix

In **Typical_Documents** table

Id should be **max id + 1 Select All, S.No., Typical Name, Typical ID, Country**

Run ··· firestop-fs-selector

c-aj-2674-images.sql M

server > src > postgres > approvals-db > c-aj-2674-images.sql

You, 23 seconds ago | 2 authors (Sadilo, Denys (LI - External) and one other)

```sql
INSERT INTO "Documents" ("Id", "ExternalId", "Type", "Name", "ExternalUrl", "MarketId")
VALUES (8176, '9669d57d-22af-4a92-b898-8db4c1ef7733', 'SystemSVG', 'C-AJ-2674.svg', 'hna/hna/CAJ2674b.svg', 2);

INSERT INTO "Typical_Documents" ("Id", "TypicalId", "DocumentId")
VALUES (8750, 1133, 8176);
```

Thank you

YOGESH GANGWAR

Hilti Aktiengesellschaft
Feldkircherstrasse 100
9494 Schaan, Liechtenstein

yogesh.gangwar@hilti.com
www.hilti.group