

## PostgreSQL FTS Relay Application

PostgreSQL FTS Relay web application is built on Spring MVC framework module. The application gets compiled on JDK 1.8 through Maven, and runs on any environment where the JDK is available. The only external dependency for the application – is external PostgreSQL database.

### Source Repository

The sources for the application are in GitHub <https://github.com/syemialy/postgres-jbi> To work with the project you may fork the repository or request private access to it.

### Configuration

Application has only single configuration file – *application.properties*, where postgres credentials are supplied to the application. When running application on various environment it is suggested to initialize postgres settings not via properties, but through the environment variables.





When compiled, the application may run via command:

```
java -Ddatabase.user=tnkxkfefnfysiw -Ddatabase.pwd=eoyUq2ZLcYcPY7U0bvxtRhLAXx  
-Ddatabase.url=jdbc:postgresql://ec2-54-225-255-208.compute-  
1.amazonaws.com:5432/d85lilb3bg1tg0 $JAVA_OPTS -jar target/dependency/jetty-  
runner.jar --port $PORT target/*.war
```

### Authentication

Basic http authentication is used to prevent unauthorized usage of the REST services. By default the application is protected by the realm where user *test* identified by password *test* is registered. To change credentials, it is suggested to use two additional environment variables: **restApiUsername** and **restApiPassword**.

When running the application on Heroku, one must change the authentication credentials via configuration variables as shown on the screenshot below, the variable values should be kept private and shared with application clients only.

restApiPassword	password	 
restApiUsername	semelianov	 

## SQL Injection Protection

The application dynamically assembles three types of statements – SELECT, CREATE INDEX, DROP INDEX. The process of creating SQL statement depends on the content of incoming messages. It is very likely that intruder may take an advantage of SQL Injection attack by sending a command of the following content to the search service

```
$ curl -X POST -H 'Content-type:application/json' -  
d'{"table":{"name":"products","columns":[{"name":"description","selectable":false,"tsinclude":true},{  
"name":"count(id)","selectable":true,"tsinclude":false},{  
"name":"product_name","selectable":false,"tsinclude":true}],  
"query":"need to connect","orderby":"product_name;dElete * FROM PRODUCTS"}'  
http://localhost:8081/srv/search
```

The application will catch such an attempt and the following error message will be thrown

```
{"error_message":"SQL Injection","error":true}
```

The error will be accompanied by the following log statement:

```
possible SQL injection attack with statement count(id) FROM products WHERE  
to_tsvector('english',coalesce(description,'') || ' ' || coalesce(product_name,'')) @@ plainto_tsquery('need  
to connect') ORDER BY product_name;dElete * FROM PRODUCTS
```

The SQL Injection protection is made using well known principles:

1. A statement is checked for having statement separator blocks, such as semicolons
2. A statement is checked for having additional SQL statements like drop, create, delete, etc.
3. A statement is checked for having special characters like \*, [], etc

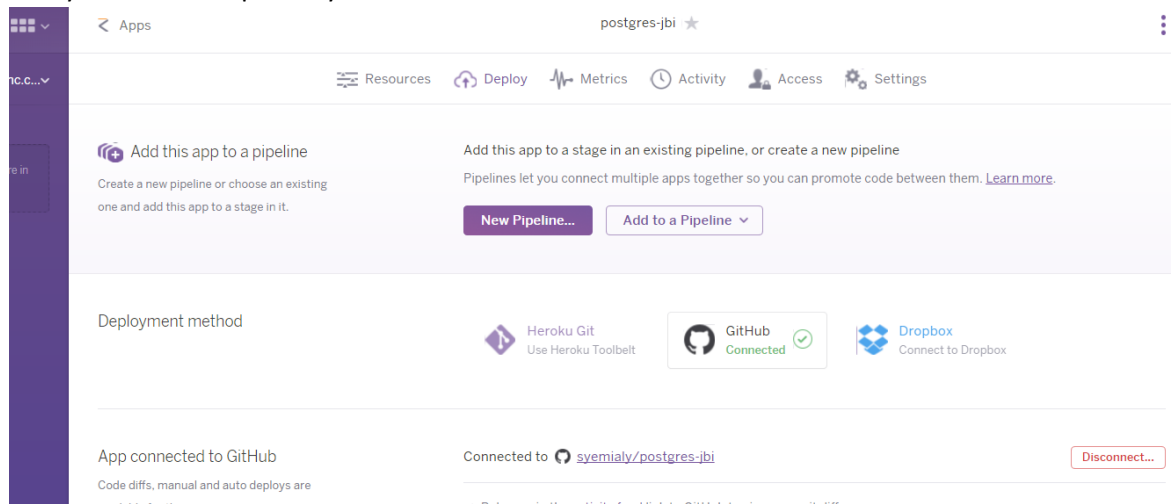
To read more on SQL Injection protection one may refer to <http://larrysteinle.com/2011/02/20/use-regular-expressions-to-detect-sql-code-injection>

## Deployment Instructions

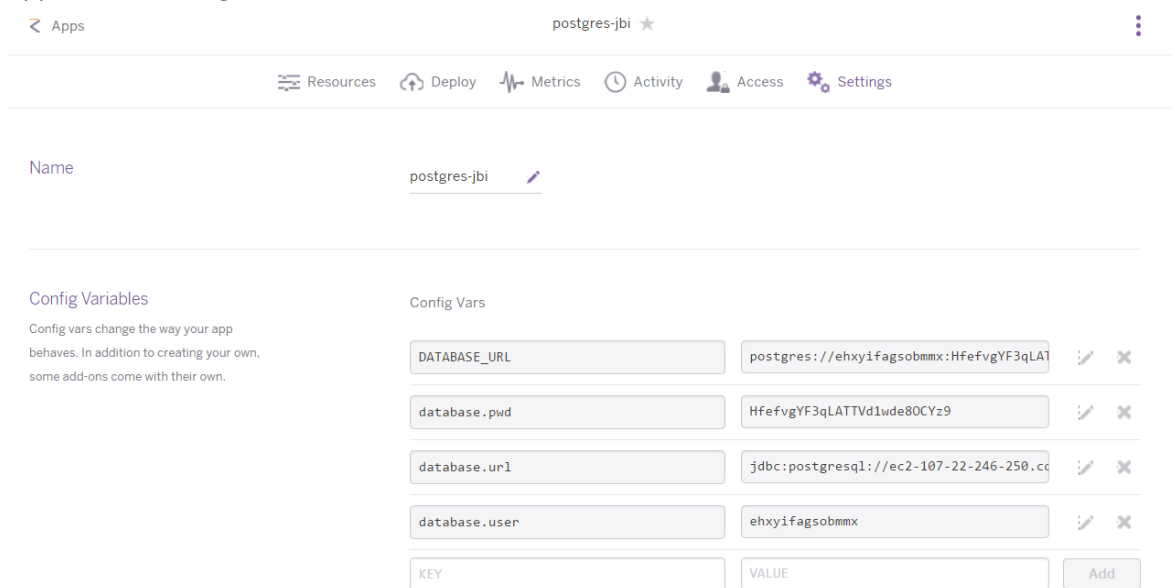
### Deployment to Heroku via GitHub

The deployment instructions assume you have forked the repository **postgres-jbi**

- Create new Heroku application
- If using external Postgres, make sure the database, attached to your dyno by default is erased and no DATABASE\_URL variable exists in your application environment.
- On the application settings page, navigate to Deploy section
  - Link your GitHub repository



- On the application Settings section, add all three environment variables .



- - Go Back to Deploy section and execute manual deploy of your new application from your github repository

#### Manual deploy

Deploy the current state of a branch to this app.

#### Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#).

 master

Deploy Branch

Once your application is deployed, you may use curl to validate all the REST services. Samples of the curl usage may be found in docs/curl.samples.txt file.

### REST Service Endpoints

Endpoint	Method	Format	Description
<b>srv/search</b>	POST	JSON	A text query passed in JSON body is used to create FTS search request against underlying PostgreSQL database
<b>srv/index</b>	POST	JSON	Create index on documents, created based on the selected columns. Columns are added to the ts_vector with coalesce() function
<b>srv/async/index</b>	POST	JSON	Creates index in asynchronous mode. The service should be used when index creation procedure takes enough time to force your synchronous http post request to time out
<b>srv/index</b>	GET	JSON	Checks if index with the exact name exists within your database
<b>srv/index</b>	DELETE	JSON	Drops the index by name

## Data Format for JSON messages

### Search Request

Field	Type	Mandatory	Description
<b>table</b>	Object	Yes	
<b>name</b>	String	Yes	Name of the table against which the FTS search will be executed
<b>columns</b>	Object[]	Yes	Column name(s) to create a document (unit of searching)
<b>name</b>	String	Yes	Name of the column within the table
<b>selectable</b>	Boolean	Yes	Indicates if column will be in SELECT statement predicate
<b>tsvectorinclude</b>	Boolean	Yes	Indicates if column will be used to create ts_vector
<b>query</b>	String	Yes	User-written text to be used in <i>plain_tsquery</i> function to create search terms
<b>orderby</b>	String	No	Specify column name(s) to order the result set
<b>limit</b>	Integer	No	Number of records to return
<b>offset</b>	Integer	No	Records offset
<b>configuration</b>	String	No	Configuration name to be used to parse and normalize strings. By default it will be set to <i>english</i>

Note: According to the documentation <http://www.postgresql.org/docs/current/static/queries-limit.html>, using LIMIT and OFFSET should be avoided due to the risk of having inconsistent results.

```
{ "table": {
    "name": "products",
    columns:[
        {"name":"product_name", "selectable":true, "tsvectorinclude":true},
        {"name":"description", "selectable":false, "tsvectorinclude":true},
        {"name":"product_id", "selectable":true, "tsvectorinclude":false}],
    "query":"sold by pair",
    "configuration":"english"
}
```

## Search Response

Field	Type	Mandatory	Description	
result	Object	Yes		
	<i>sqlstatement</i>	String	Yes	Name of the table against which the FTS search will be executed
	<i>records</i>	Object[]	Yes	
	<i>javatimemls</i>	Integer	Yes	Indicates time the java program ran FTS search, this is to collect statement statistics
error	Boolean	Yes	User-written text to be used in <i>plain_tsquery</i> function to create search terms	
error_message	String	No	Error message	

## Create Index Request

This is the same format for `srv/async/index` and `srv/index` endpoints

Field	Type	Mandatory	Description
<b>name</b>	String	Yes	Name of the index to be created
<b>type</b>	String	No	Type of the index to be created, GIN will be default type.
<b>table</b>	Object	Yes	Name of the table for which index will be created
<b>name</b>	String	Yes	Name of the table against which the FTS search will be executed
<b>columns</b>	Object[]	Yes	Column name(s) to create a document (unit of searching)
<b>name</b>	String	Yes	
<b>configuration</b>	String	No	Configuration name to be used to parse and normalize strings. By default it will be set to <i>english</i>

```
{"name":"idx_prddescr",  
  "table":{"name":"products",  
    "columns":[{"name":"description"}, {"name":"product_name"}]}}
```

### Create Index Response (Synchronous)

Field	Type	Mandatory	Description
<b>result</b>	String	Yes	Status of the request
<b>sqlstatement</b>	String	Yes	The SQL statement which was executed
<b>error</b>	Boolean	Yes	User-written text to be used in <i>plain_tsquery</i> function to create search terms
<b>error_message</b>	String	No	Error message

```
{"result":"created",  
  "error":false,  
  "sqlstatement":"CREATE INDEX idx_prddescr ON products USING GIN  
(to_tsvector(\u0027english\u0027,coalesce(description,\u0027\u0027) || \u0027\u0027 ||  
coalesce(product_name,\u0027\u0027)))"  
}
```

### Create Index Response (Asynchronous)

Field	Type	Mandatory	Description
<b>result</b>	String	Yes	Status of the request
<b>message</b>	String	No	Message for the request
<b>error</b>	Boolean	Yes	User-written text to be used in <i>plain_tsquery</i> function to create search terms
<b>error_message</b>	String	No	Error message

```
{"result":"scheduled","error":false,"sqlstatement":"check later for index create status"}
```

### Drop Index Request / Check Index Request

Field	Type	Mandatory	Description
<b>name</b>	String	Yes	Name of the index to be dropped

```
{"name":"idx_prddescr"}
```

### Drop Index Response

Field	Type	Mandatory	Description
<b>result</b>	String	Yes	Status of the request
<b>sqlstatement</b>	String	Yes	The SQL statement which was executed
<b>error</b>	Boolean	Yes	User-written text to be used in <i>plain_tsquery</i> function to create search terms
<b>error_message</b>	String	No	Error message

```
{"result":"dropped",  
  "error":false,  
  "sqlstatement":"DROP INDEX idx_prddescr"}
```

## Check Index Response

Field	Type	Mandatory	Description
<b>result</b>	String	Yes	Status of the request
<b>location</b>	String	Yes	Table location where index belongs
<b>error</b>	Boolean	Yes	User-written text to be used in <i>plain_tsquery</i> function to create search terms
<b>error_message</b>	String	No	Error message

```
{"result":[{"location":"products.idx_ts3"}],"error":false}
```