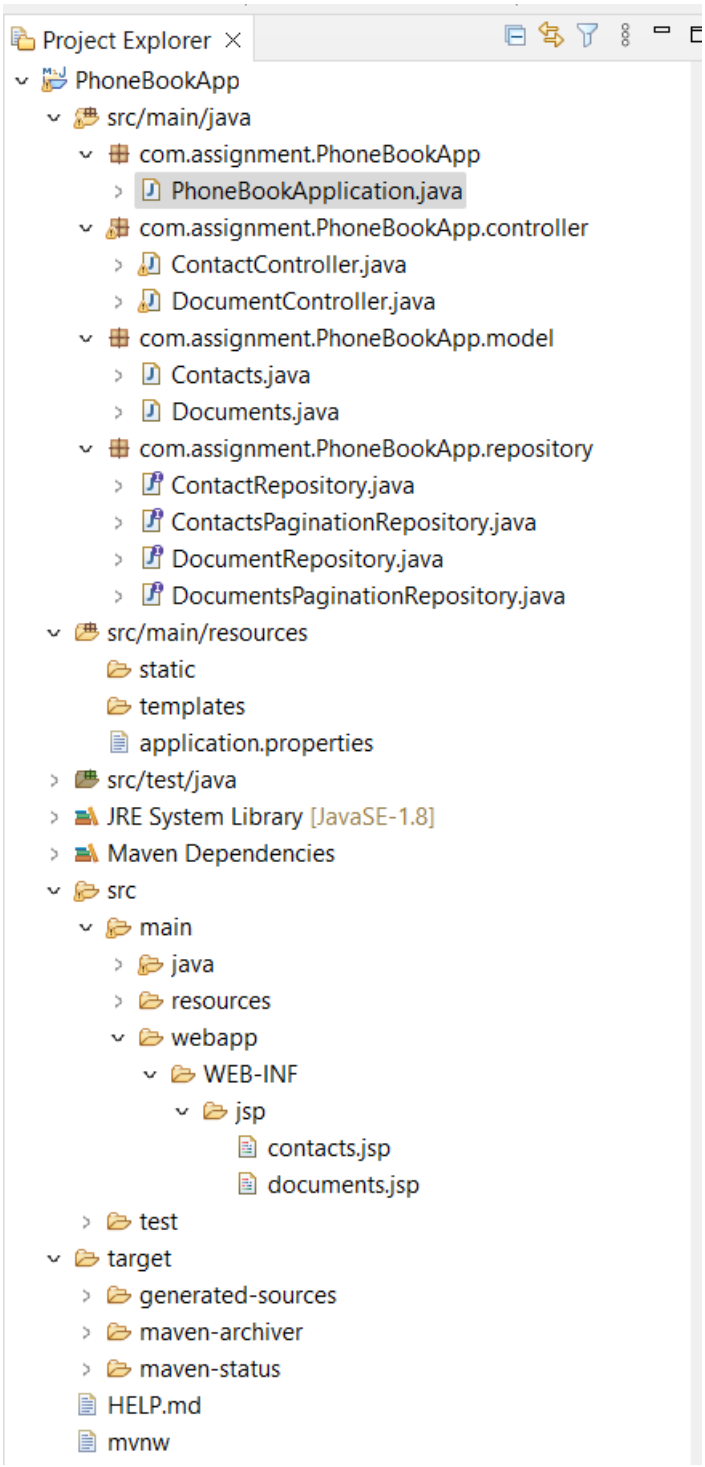
Name : SAHANA BHAT Date : 13/04/2023

MPockket_Both Assignment_Documentation – PhoneBook & Search App

Execution:

1. Download the project from github or find the attached zip – SpringBoot Web Application : **PhoneBookApp**
2. **Note I have implemented both the assignment phonebook and the search app within this one project which are running on two different end point ie phone book on /api/contacts and search app on /api/documents**
3. For Spring Boot Web application I have used Springboot with the dependencies mentioned in question, Spring boot with JSP View, MySQL RDBM as Data Store, JPA and PaginationSort Repositories, Bootstrap with JSP, CSS styles etc
4. While running the project Internet is must as I am calling online Javascript library classes such as jquery.min.js and other css style links
5. Import the Spring boot web app into Eclipse (which I used) -> Maven Update Project -> Maven clean -> Maven Build -> Run PhoneBookApplication.java as Java Application.
6. Then go to http://localhost:8080/

7. Following is the Project folder structure with controller, entity, repositories and a JSP



8. Please find the mysql dump to quickly import.

phonebook.sql.txt

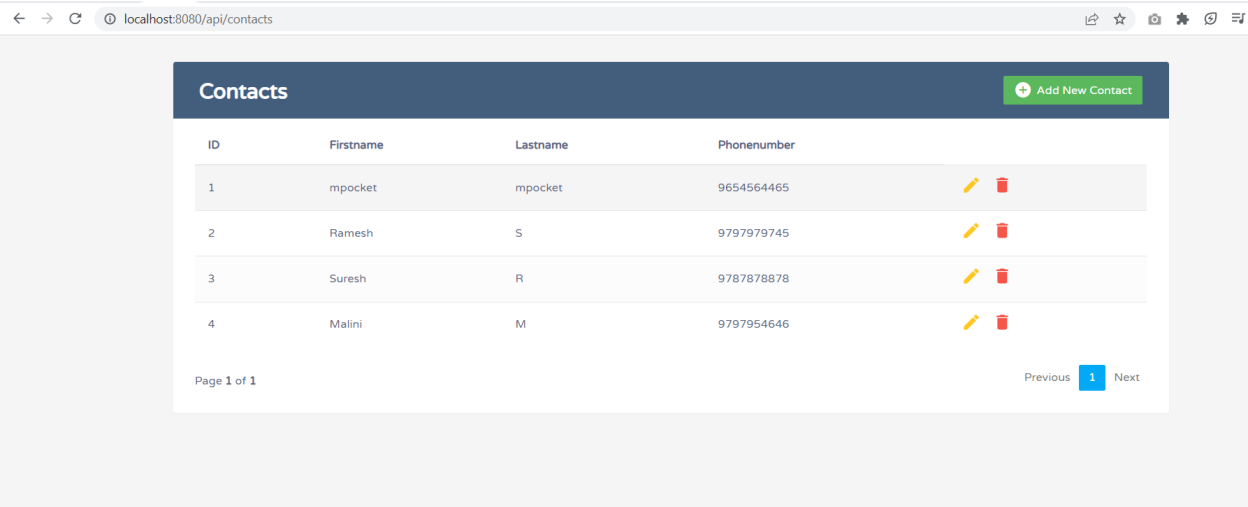
9. Following are the rest end points used in this spring web app

a. listing of contact (with pagination support) => GET /api/contacts
b. create new contact => POST /api/contacts
c. update contact => POST /api/contacts/{id} /edit
d. delete contact => POST /api/contacts/{id}/delete

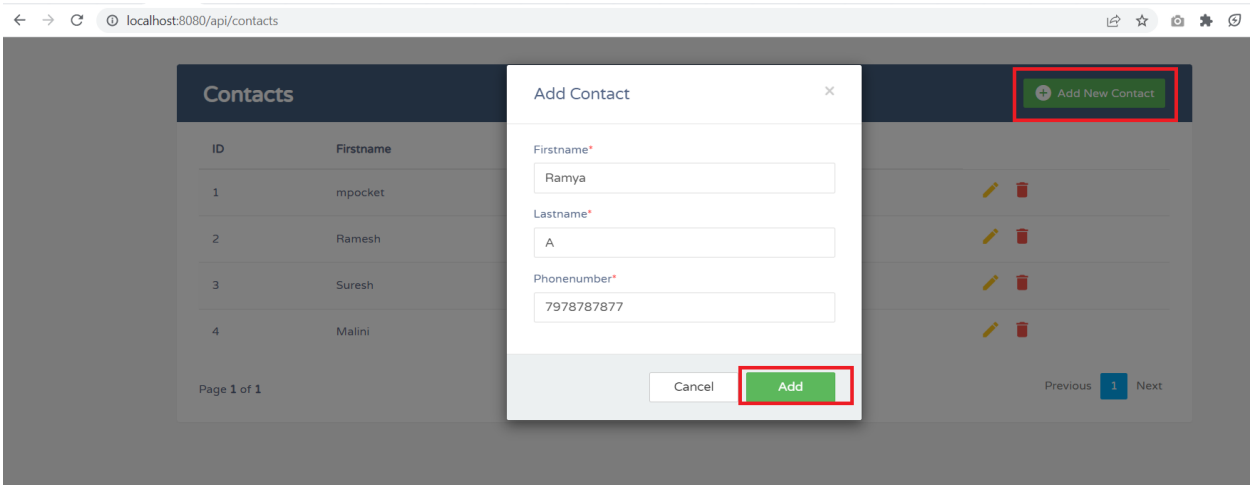
Note: As HTML/JSP won't support forms submission via put/delete method so changed the api design slightly and used post instead for edit and delete. But in the advance task for Creating Rest controller I have used GET,POST,PUT,DELTE with Request and Responses are in JSON, The respective info/screenshot will be added later in this doc

Output screenshots of spring boot webapp- Phonebook

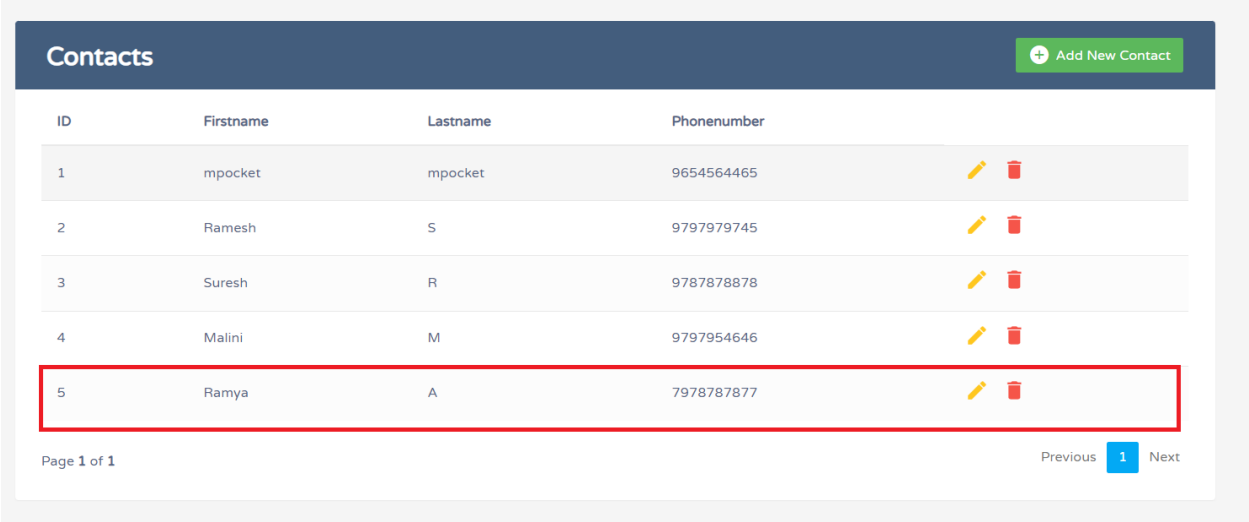
1. Home Page with list of all contacts from database with Add contact, Edit, Delete and Pagination functionalities



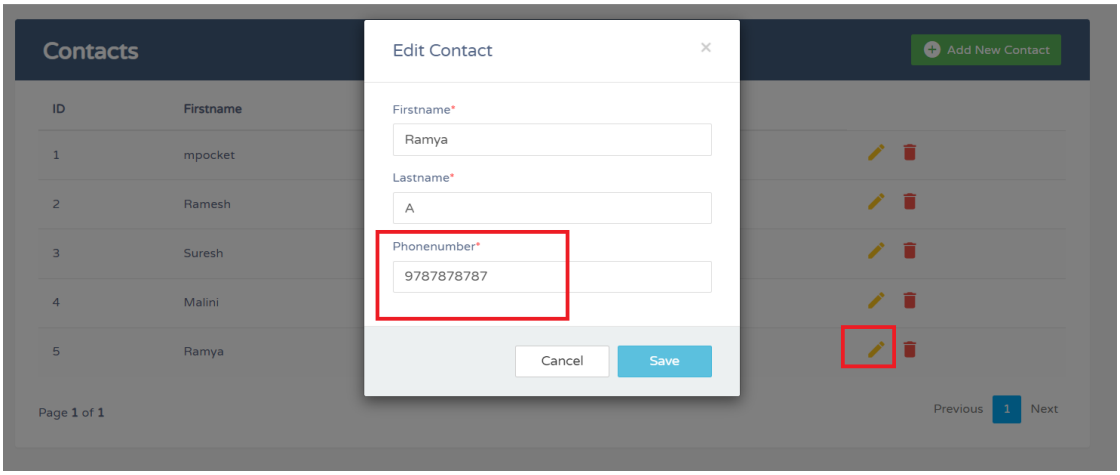
2. Add Contact Modal

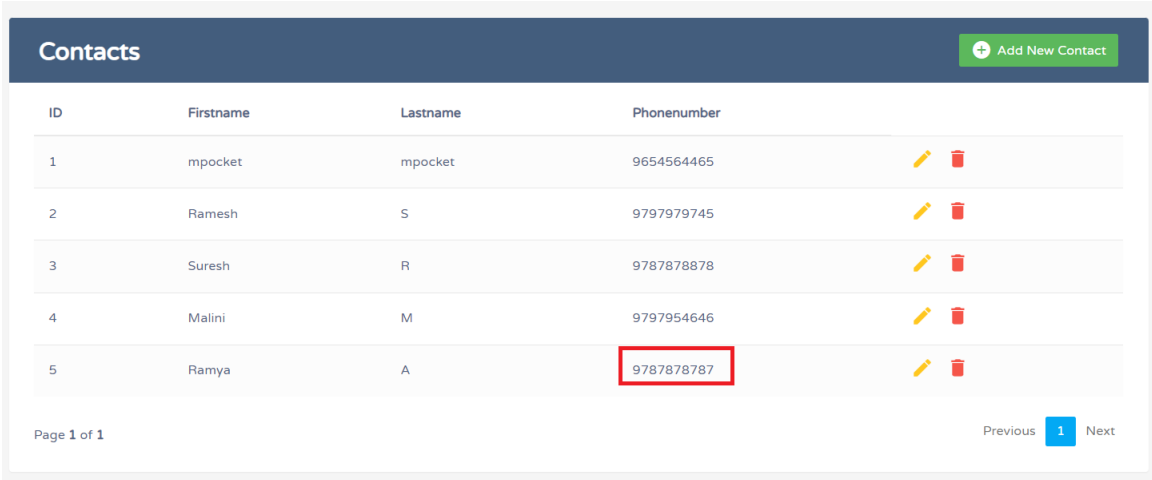


The Data Added for Ramya contact is populated

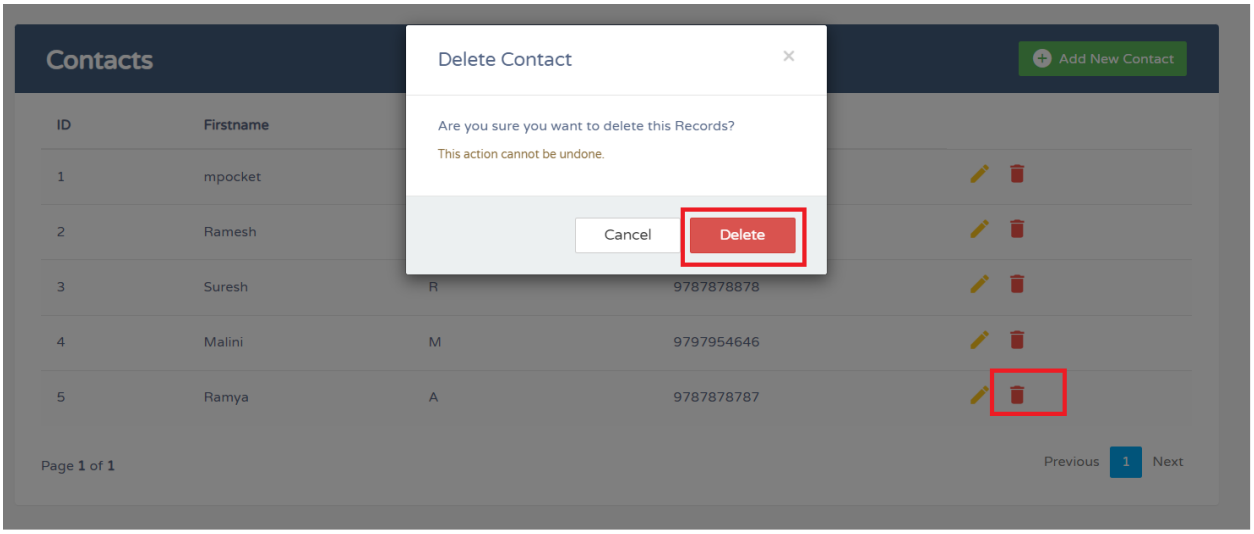


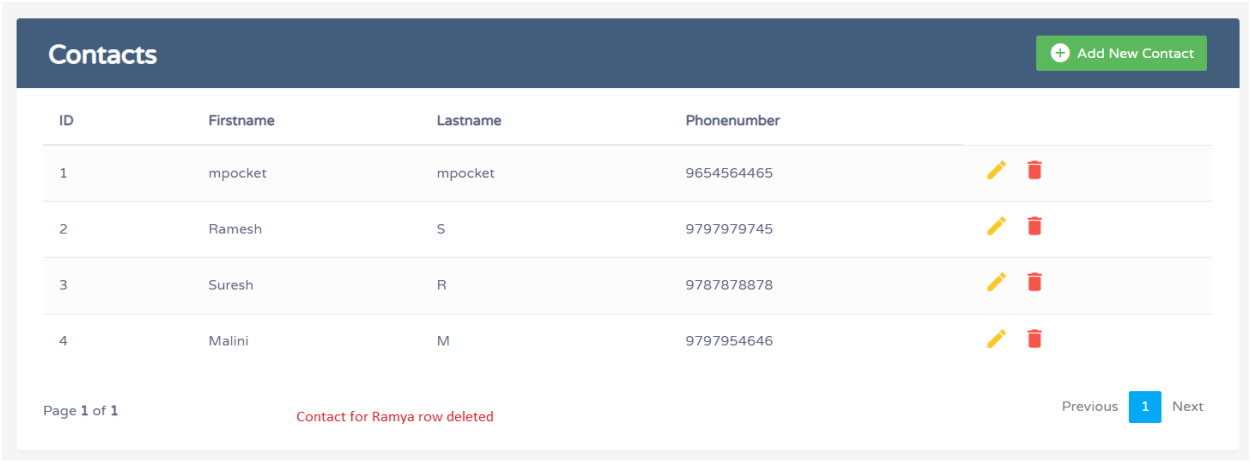
3. Edit Contact for Ramya





4. Delete contact for Ramya





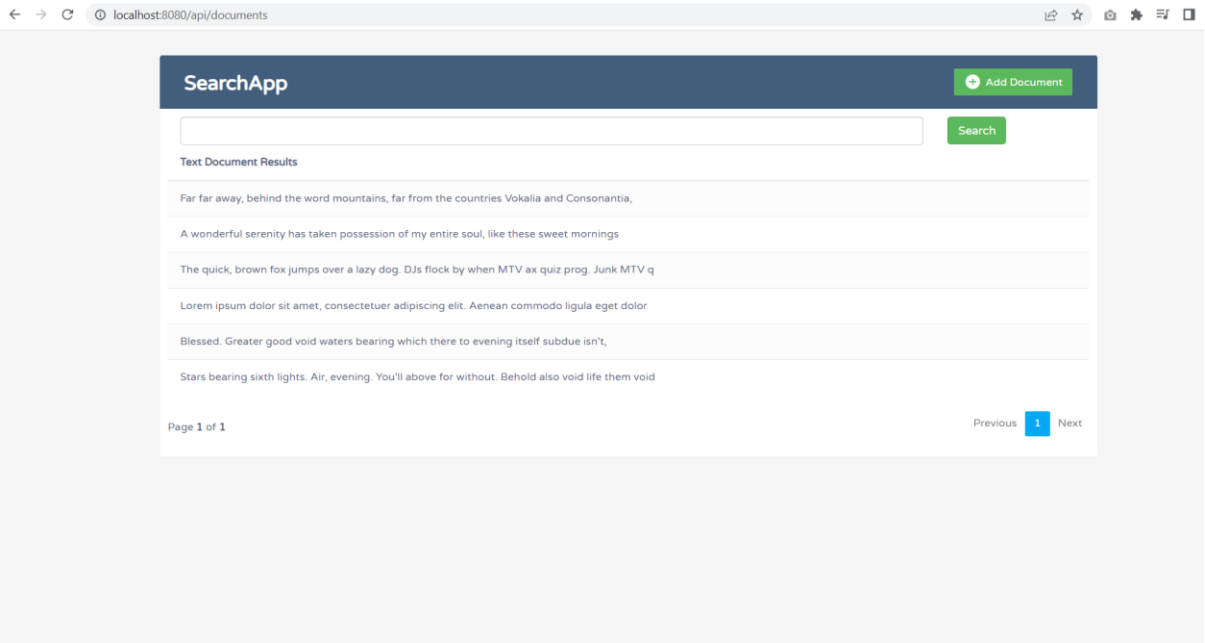
SEARCH APP ASSIGNMENT

How Spring boot Cache works

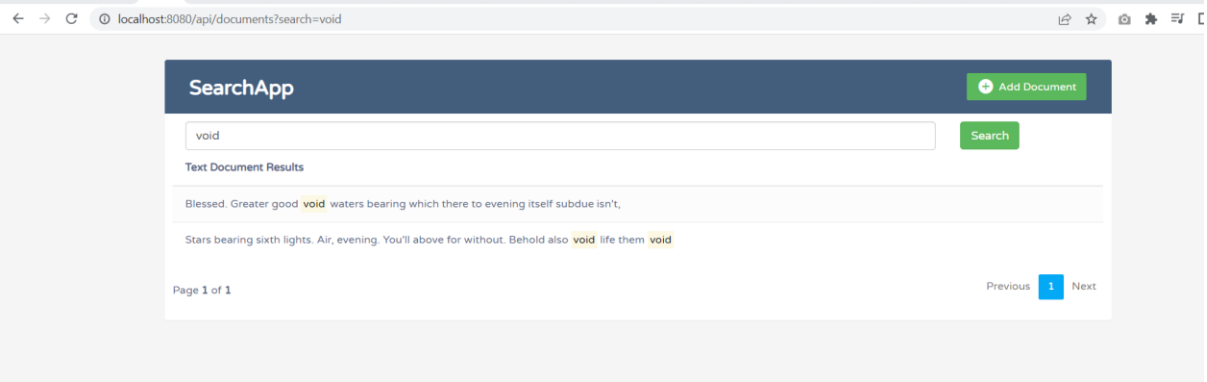
1. Hit the GET api end point , this will fetch the data freshly from db first time and stores the result in cache
2. Hit the Get api end point,it will fetch from cache
3. Update the db (save or savorupdate methods) then cache will be cleared and ready to fetch fresh data from db

Search app flow

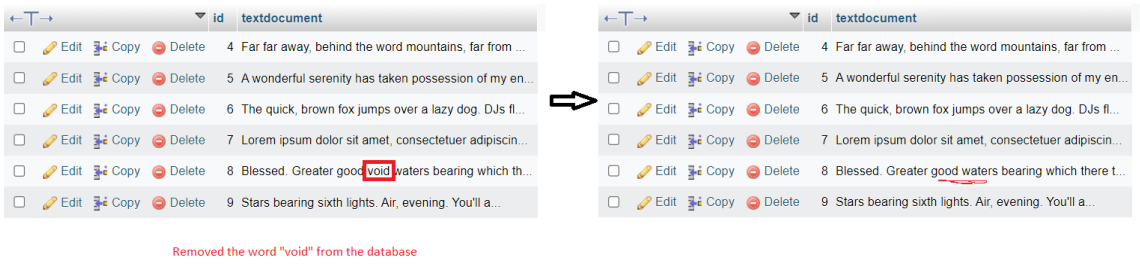
Simple search app home page , api end point is “/api/documents” – all the text documents added in the db are populating and we have option to add new text by clicking Add document and add it.



Searched key word “void” showing two records. As this is first time searching we get the data directly from the db and stores the result in cache for next time quick fetching the same result, this data will not change until we modify anything in the db via the spring boot web application

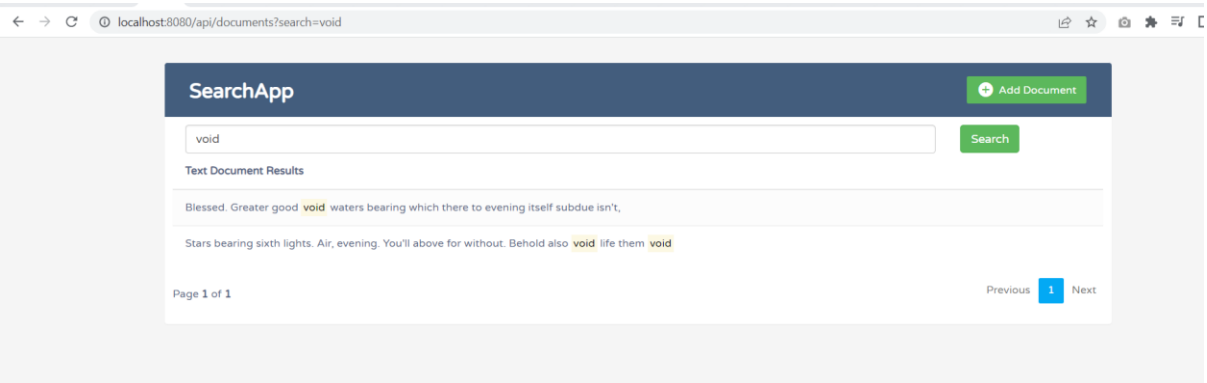


To test the cache working or not, lets modify manually the db directly from mysql viewer, removing the keyword void from the id 8 as shown below image and the db record for 8th id looks as shown in the image below (right). Note here we are changing manually the records and we should get the result same as previous because cache has not cleared, the cache will only clear when we update the db via the spring boot web application

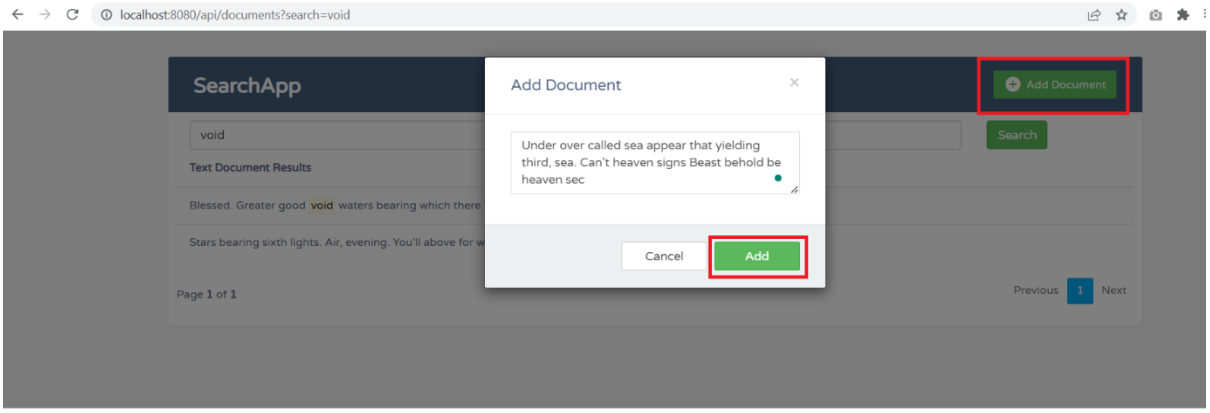


Removed the word "void" from the database

After modifying manually the db, the result shows from the cache. 2 records same as first time fetching even though manuaaly we have changed db , this is because it is fetching from cache



Lets add new record from spring boot web application which internally updates the db and hence clears the cache freshly



As cache got cleared because of previous db update from the spring boot web application the same keyword search shows only one record