REST API WITH SPRING BOOT

Design Diagram

<<Java Class>>
ⒼContactController
com.solstice.api.contact.controller

§ᵍᶠ logger: Logger

- ⚲ ContactController()
- ● listAllContacts():ResponseEntity<List<Contact>>
- ● deleteAllContacts():ResponseEntity<Contact>
- ● deleteContactByID(Long):ResponseEntity<Contact>
- ● searchContactsByEmail(String):ResponseEntity<List<Contact>>
- ● searchContactsByCity(String):ResponseEntity<List<Contact>>
- ● addContact(Contact):Contact
- ● addContact(String):Contact

<<Java Class>>
ⒼContact
com.solstice.api.contact.entity

- ▫ id: Long
- ▫ firstname: String
- ▫ lastname: String
- ▫ email: String
- ▫ phonenumber: String
- ▫ image: byte[]
- ▫ city: String
- ▫ state: String
- ▫ country: String
- ▫ zip: String
- ▫ address: String

- ⚲ Contact()
- ● getFirstname():String
- ● setFirstname(String):void
- ● getLastname():String
- ● setLastname(String):void
- ● getCity():String
- ● setCity(String):void
- ● getState():String
- ● setState(String):void
- ● getCountry():String
- ● setCountry(String):void
- ● getZip():String
- ● setZip(String):void
- ● setPhonenumber(String):void

~contactService | 0..1

<<Java Interface>>
ⓘContactService
com.solstice.api.contact.service

- ● findAllContacts():List<Contact>
- ● deleteAllContacts():void
- ● deleteContactByID(Long):void
- ● addContact(Contact):Contact
- ● getContactById(Long):Contact
- ● searchContactsByEmailorPhone(String):Iterable<Contact>
- ● searchContactsByCityOrState(String):Iterable<Contact>

<<Java Class>>
ⒼContactServiceImpl
com.solstice.api.contact.service

- ▫ contactDAO: IContactDAO

- ⚲ ContactServiceImpl()
- ● findAllContacts():List<Contact>
- ● deleteAllContacts():void
- ● deleteContactByID(Long):void
- ● searchContactsByEmailorPhone(String):List<Contact>
- ● searchContactsByCityOrState(String):List<Contact>
- ● addContact(Contact):Contact
- ● getContactById(Long):Contact

-contactRepository

0..1

<<Java Interface>>
ⓘContactRepository
com.solstice.api.contact.repository

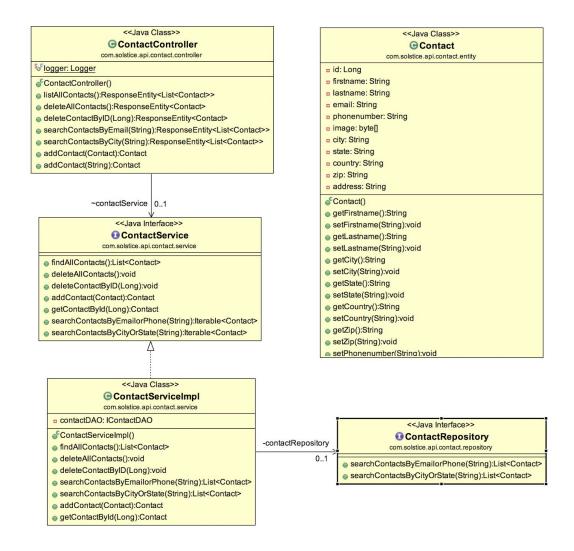- ● searchContactsByEmailorPhone(String):List<Contact>
- ● searchContactsByCityOrState(String):List<Contact>
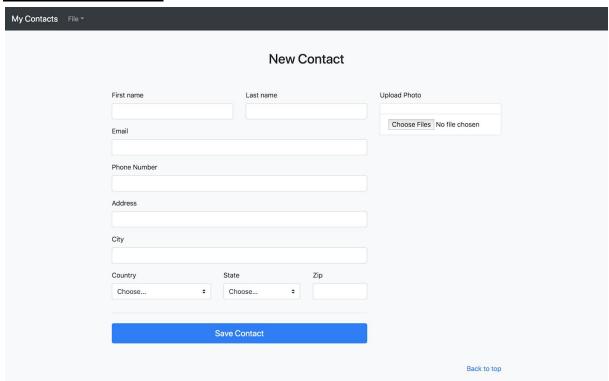
EASY SETUP INSTRUCTIONS

- Create Spring Initializer Project at https://start.spring.io/
- Provide Group and Artifact as com.solstice and contact
- Unzip the artifact and create existing Maven Project in Eclipse
- Copy the src content from the source Zip code and pom.xml
- Modify the application properties for database credentials and access
- Change the ddl property to create/none appropriately
- The Server will be run on localhost:8080 while running the maven build
- The welcoming html shows some of the api functionalities
- Alternatively run mvn package and generate the jar file which can be easily deployable in AWS
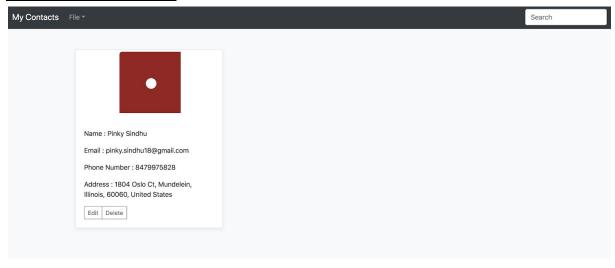
UNIT TEST CASE LOCATIONS
- src/test/java

UI SCREENS

Add/Edit Contact Screen

List/Delete/Search Screen

REST API ENDPOINTS

Content-Type = application/json

1. To find the contact by ID
http://192.168.0.105:8080/api/contact/{id}

Request : GET : http://192.168.0.105:8080/contact/1

2. To find the list of all Contacts
http://192.168.0.105:8080/api/list/

Request : GET : http://192.168.0.105:8080/api/list/
Response
[{"id":4,"firstname":"Jane","lastname":"Doe","email":"jane.doe@gmail.com","phonenumber":"
12345678","image":"123","city":"Chicago","state":"Illinois","country":"United
States","zip":"54321","address":"128 Lans Lane"}]

3. To Delete all Contacts
http://192.168.0.105:8080/api/delete

Request : DELETE : http://192.168.0.105:8080/api/delete

4. To delete contact by Id
http://192.168.0.105:8080/api/delete/{id}

Request : GET : http://192.168.0.105:8080/api/delete/1

5. To search contact by Phone Number/Email
http://192.168.0.105:8080/api/search/{value}

Request : GET : http://192.168.0.105:8080/api/contact/pinky@email.com

6. To retrieve contact by State or City
http://192.168.0.105:8080/api/retrieve/{value}

Request : GET : http://192.168.0.105:8080/api/contact/Illinois

7. To Save a Contact
http://192.168.0.105:8080/api/

Request :
{
"id":4,
"firstname":"Jane",

"lastname":"Doe",
"email":"jane.doe@gmail.com",
"phonenumber":"12345678",
"image":<base64 data>,
"city":"Chicago",
"state":"Illinois",
"country":"United States",
"zip":"54321",
"address":"128 Lans Lane"
}