**APNESATHI BACKEND DOCUMENTATION FOR WEB PORTAL**

**BACKEND CODE SETUP:**

**Software’s required:**

* Spring Tool Suite 4 / Eclipse IDE
* Postman
* My SQL Workbench 8.0 CE

**For MySQL setup:**

Create connection using the following details (DEV Server):

* spring.datasource.url=jdbc:mysql://15.207.42.209:3306/KEFUSER
* spring.datasource.username=apne\_saathi\_dev
* spring.datasource.password=InfyDev2020!

(Prod Server):

* spring.datasource.url=jdbc:mysql://3.7.39.123:3306/KEFUSER
* spring.datasource.username=apne\_saathi\_dev
* spring.datasource.password=InfyDev2020!

**For Backend Setup:**

* File ->Import->Git->Clone By URI
* For Authentication you can give your personal git account id and password.
* Once the project is imported, you’ll find ApneSaathiBackend ->Right Click->Configure->Convert into Plugin project)
* In ApneSaathiBackend->developer\_parvathy->VolunteerApp ->Run As Maven Build(-X clean install)
* You can find Backend Documentation for Android App API in **src/main/resource** in git.

**WAR deployment steps on the server:**

* Import the latest code from GitHub in the Eclipse/Spring Tool Suite IDE.
* Run as Maven Build (-X clean install). Built should be successful
* Find war in the given path ApneSaathiBackend->developer-parvathy-> VolunteerApp->target
* Select Volunteer.war. inside target folder
* Rename the war to Volunteer.war
* Deploy the war on the server.
* If deployment is successful running status should be true. For errors during deployment check log files.

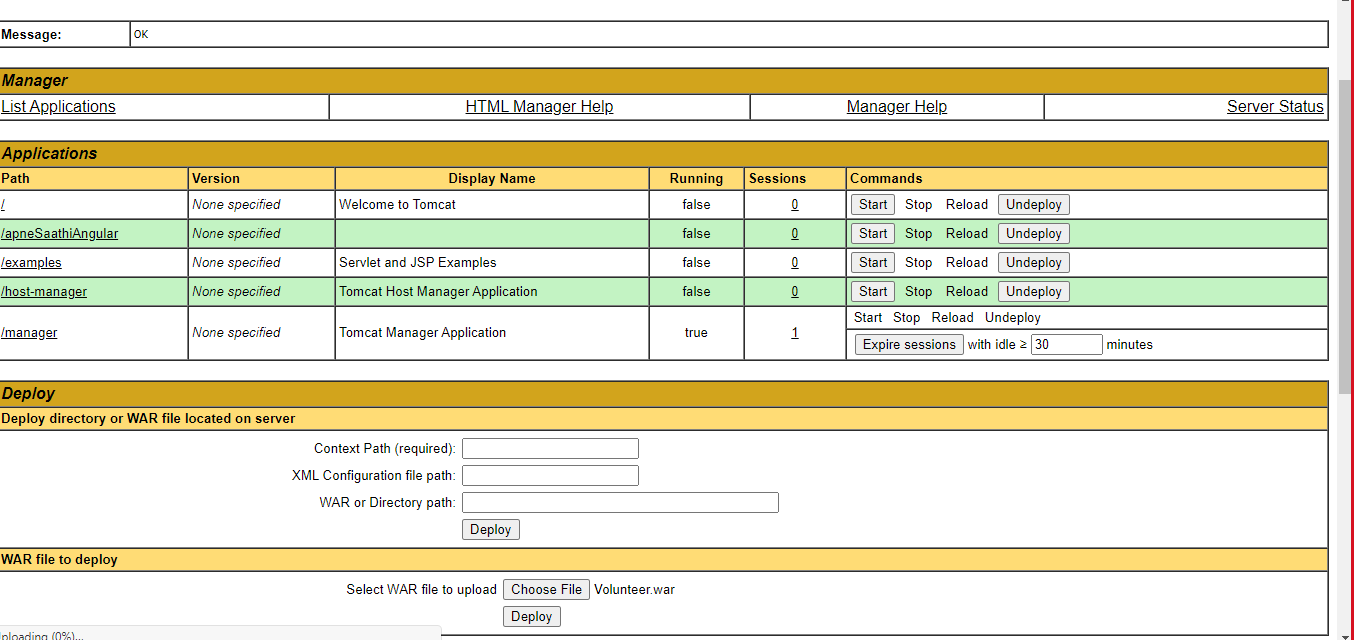
**To Deploy the code on the Production server:**

Figure : War deployment

* Take backup of data from all tables.
* Run the database scripts on the development server.
* Insert the data into the tables again.
* Change the database connections to the production database in the **application.properties** file
* Use Maven Clean Build on the project
* Deploy the Volunteer.war into the server
* Postman Link for dev server (Portal):

[**https://www.getpostman.com/collections/5ed937bf457eca4e64a0**](https://www.getpostman.com/collections/5ed937bf457eca4e64a0)

* Postman Link for dev server (Portal and App):

[**https://www.getpostman.com/collections/b943bad17e19bf7d4e6a**](https://www.getpostman.com/collections/b943bad17e19bf7d4e6a)

**Please find the API list below which we are using for Backend portal.**

* /getVolunteersList

This API is used to display list of Volunteers. In request body we pass the following parameters:

status:Active /Deboarded

filterState:

filterDistrict:

filterBlock:

sortBy: rating/assignedSrCitizen

sortType: asc,desc,

excludeIds:[]

Exclude ids is used to exclude the volunteer from that volunteer list

* /getSrCitizenList

This API is used to display list of Senior Citizen. In request body we pass the following parameters:

status: Assigned/ UnAssigned /Deboarded

filterState:

filterDistrict

filterBlock

In response we will receive the list of Senior Citizens based on the filter criteria

* /assignSrCitizen

In this API we assign SeniorCitizens to Volunteers. For request body we are passing volunteer id, role, adminId and senior citizen list that needs to be assigned to a particular volunteer.

* /distributeSrCitizen

This API is used for distributing Senior Citizens equally among volunteers while deboarding the volunteer. For request body we are passing list of volunteers to whom the senior citizen should be assigned.

* /getVolunteerDetails

This API is used to display the details of a volunteer. For request body we are passing id of the volunteer.

* /transferVolunteer

This API is used to transfer state, district or block of a volunteer. In request body we are passing the volunteer id, state, district ,block of volunteer.

* /deboardVolunteer

This API is used to deboard a volunteer. In request body pass the id of volunteer which is to be deboarded. The status of the volunteer should change to Deboarded in the database.

* /UnassignSrCitizen

This API is used to unassign the Senior Citizens. In request body pass the Senior Citizen list and volunteer id to which they are currently assigned.

* /srCitizenByVolunteer

This API is used to get the list of Senior Citizen assigned to a volunteer. In request body we pass the id of volunteer and in response we get the list of senior citizen assigned to that volunteer.

* /srCitizenPersonalinfo

In this API we pass the senior citizen id to get the personal details along with details of medical and grievance details of that senior citizen.

* /srCitizenQueries

This API is used to get the list of senior citizen queries based on query Type, state district, block .Sort by can be applied to the list.The request body has following parameters:

queryType : in progress / pending / resolved

sortBy : priority, createdDate

sortType: asc,desc

* /updateGreivanceStatus

This API is used to update the grievance status for a grievance. For request body we are sending trackingId, status, description, reivewedBy.

status can be RESOLVED/ UNDER REVIEW/ RAISED

* /deboardSrCitizen

This API is used to deboard a senior citizen. In request body we pass the id of senior citizen, phoneNo of the senior citizen and deboardReasons.

* /verifyAdmin

This API is used to login in the backend portal. In request body we are passing userName and password. If the username and password is correct, then the user will be able to login otherwise appropriate error message will be displayed.

* / getInProgressQueryDetails

This API is used to view the details of in progress queries. The Request body will have issueId and status:” in progress”.

* /importVolunteers

This API is used to import Volunteers into the database from the csv file. The request body has following parameters:

file : multipart .csv file

adminId :

adminRole:

The response will contain:

uploadFileName : file name of the uploaded file

uploadFileDownloadUrl : The URL where the file is getting stored. We can download the file by clicking this URL in the browser.

errorFileName : If any entries are invalid then it will generate error file with this filename

errorFileDownloadUrl : The URL where the error file is getting stored. We can download the file by clicking this URL in the browser.

successCount: No of rows successfully inserted into the database

failureCount : No of rows that have error

message : Success/Failure/Partial Success

statusCode : 0/1

* /importSrCitizen

This API is used to import Volunteers into the database from the csv file. The request body has following parameters:

file: multipart .csv file

The response will contain:

uploadFileName : file name of the uploaded file

uploadFileDownloadUrl : The URL where the file is getting stored. We can download the file by clicking this URL in the browser.

errorFileName : If any entries are invalid then it will generate error file with this filename

errorFileDownloadUrl : The URL where the error file is getting stored. We can download the file by clicking this URL in the browser.

successCount: No of rows successfully inserted into the database

failureCount : No of rows that have error

message : Success/Failure/Partial Success

statusCode : 0/1

**Package Structure:**

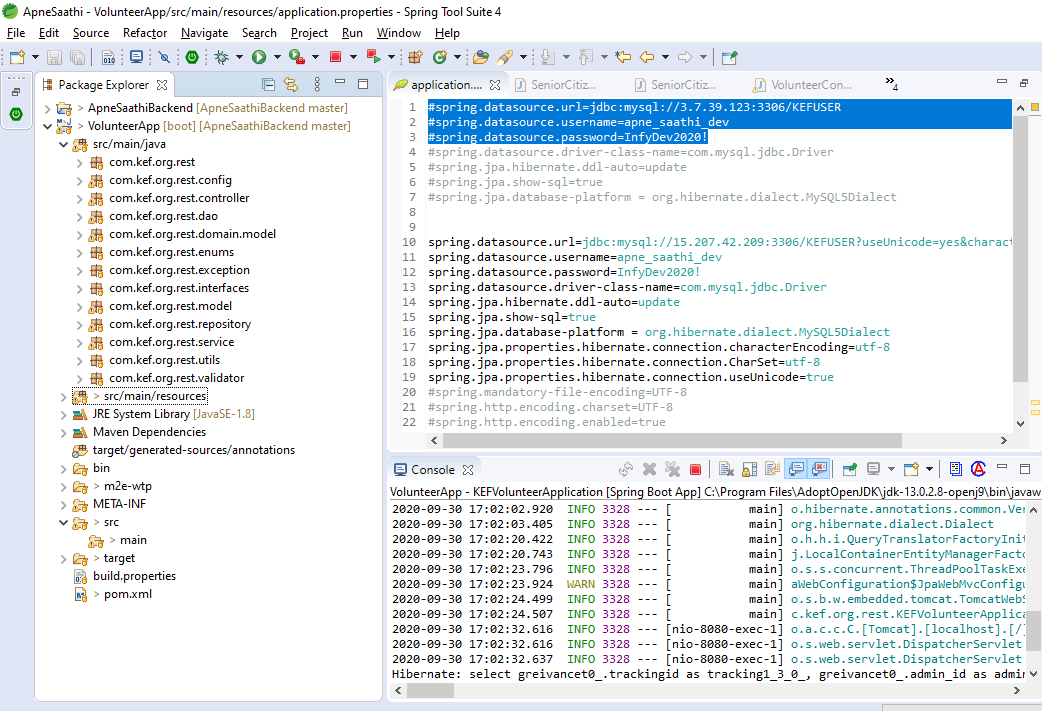


Figure : Package structure