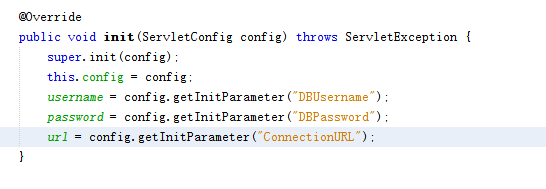
Back-end Code Document

* **General Description**

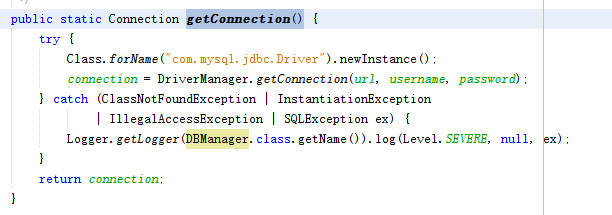
The total back-end program can be divided into four main parts, which are database connection management class, data access object, Java bean and servlets. Database connection management class is responsible for the creation, management and destruction of database connection. In data access object, we can develop functions to manipulate and query the data in database. In our project, Java bean is the user class which is used to define fields of each user. The last part servlets are used to handle different http request from front-end.

* **Database Connection Management Class**
* Class name: DBManager
* Functions:

1. init(): configure the database connection parameter such as username, password and connection URL.



1. getConnection(): get database connection object.

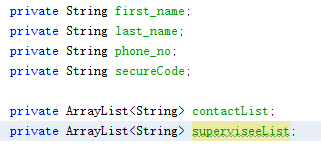


1. closeAll(): close all database connection.

* **Data Access Object:**
* Class name: UserDAO
* Functions:

1. queryUser(): query user with the phone number. If there is no such record, return null. Otherwise, return a user object.
2. addUser(): query if there exists such user before. If no such user exists, create a new user record with phone number. If record is added successfully, return true; otherwise, return false;
3. signUp(): Complete the newly created user’s information (including the user name and secure number). If update successfully, return true; otherwise, return false.
4. addEmergency(): Update(add) the emergency contact information in current user record. Each emergency contact record is expressed as a string. String format: first name\_ last name: phone number. If update successfully, return the updated user object; otherwise, return null.
5. deleteEmergency(): Update(delete) the emergency contact information in current user record. If update successfully, return true; otherwise, return false.
6. queryEmergency(): Query the emergency contact information with the user phone number. If there exists such user, return a string which includes the information of all emergency contact (split with “;”); otherwise, return null.
7. querySupervisee(): query who can be monitored by this user using user phone number. If there exists such user, return a arraylist which contains all supervisees; otherwise return null.
8. updateRoute(): Update the departure and destination information of the specific user (including the departure name, destination name, departure location and destination location). If update successfully, return true; otherwise, return false.
9. queryRoute(): query the the departure and destination information of the specific user with user phone number(including the departure name, destination name, departure location and destination location). If there exists such user, return a string which includes departure and destination information (string format: departure+"\_"+destination+"\_"+departure location+"\_"+destination location); otherwise, return null.
10. updateLocation(): Update the user location of specific user (including latitude and longitude). If update successfully, return true; otherwise, return false.
11. queryLocation(): Query the user location of specific user with phone number (including latitude and longitude). If there exists such user, return a string (string format: longitude +"\_"+latitude); otherwise, return null.

* **Java Bean:**
* Class name: User
* Functions: setters and getters.
* Fields:

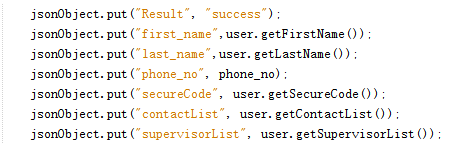


* **Servlets:**
* SignUpServlet:

Get phone number parameter from the http request and check if the user with this phone number exists. If no such user, the addUser() function in UserDAO will be called.

* LoginServlet:

Get phone number parameter from the http request and check if the user with this phone number exists (call queryUser() function in UserDAO). Then, the back-end will return a json object which contains all the user information.



* CompleteInfoServlet:

Get username, phone number and secure number from the http request and update user information in database by calling signUp() function in UserDAO.

* AddEmergencyServlet / DeleteEmergencyServlet:

Get the user phone number and emergency contact phone number from http request and update the user emergency contact name by calling addEmergency() and deleteEmergency() function in UserDAO.

* ShowEmergencyServlet / ShowSuperviseeServlet:

Get the user phone number from http request and return the emergency/ supervisee information stored in a json object

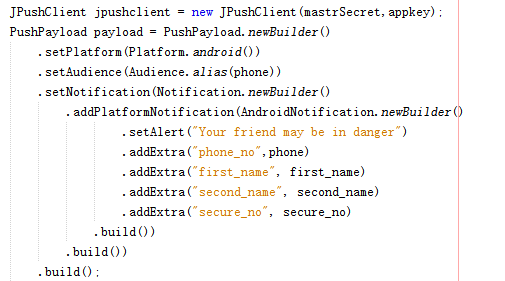




* PushServlet / ArriveServlet / DangerServlet:

Configure the notification push parameter. Each user account has an alias which is set as its own phone number. Thus, the back-end can push notification via alias. We use Jiguang Push SDK in the back-end and we need to configure the app key and master secret to ensure the back-end only work for our application.



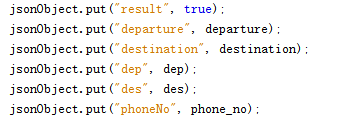


The code above showed how can we build a payload which will be pushed.

These three servlets are used to notify the emergency contact their friends start a single trip, arrive safely and meet danger respectively.

* SavePositionServlet/ PostPositionSevlet:

The first servlet is will get departure and destination information from http request and update the current user record by calling updateRoute() function in UserDAO. The second servlet will return the departure and destination information of a specific user (see the figure below).



* PositionServlet/ GetPosSevlet:

The first servlet will get the user location from http request and update the user location (parameter: longitude and latitude) in current record in database by calling updateLocation() in UserDAO. The second servlet will return a json object which includes the user location to the front-end (see below).

