JAVA SERVLET

Using SUSI.AI Accounting Model to store Device information on Server

Just like with Google Home devices and Amazon Alexa devices, SUSI Users should have a way to add and store the information of their devices (Smart Speakers) on SUSI Server, so that it could be displayed to them on various clients. Hence, we need a Servlet which could add and store the User data on the Server.

Implementation of Servlets in SUSI.AI

All servlets in SUSI extend AbstractAPIHandler class and implement APIHandler. All servlets have 4 methods, which we overwrite depending on what we want the servlet to do. They are as follows :

```
@Override
    public String getAPIPath() {
        return null;
    }

@Override
    public BaseUserRole getMinimalBaseUserRole() {
        return null;
    }

@Override
    public JSONObject getDefaultPermissions(BaseUserRole baseUserRole) {
        return null;
    }

@Override
    public ServiceResponse serviceImpl(Query post, HttpServletResponse response,
Authorization rights, JsonObjectWithDefault permissions) throws APIException {
        Return null;
    }
```

How these 4 methods work together ?

1. First method is getAPIPath(). It returns the endpoint of the servlet.
2. The second method is getMinimalBaseUserRole(). It returns the minimum privilege level required to access the endpoint.
3. The third method is getDefaultPermissions(). It gets the Default Permissions of a UserRole in SUSI Server. Different UserRoles have different permissions defined in SUSI Server.
4. Whenever the endpoint defined in the getAPIPath() method is called properly, it responds with whatever is defined in the fourth method, which is serviceImpl().

How is User data stored on SUSI Server ?

Before we move on to the actual implementation of the API required to store the device information, we should get a brief idea of how exactly User data is stored on SUSI Server.

Every SUSI User has an Accounting Object. It is a JSONObject which stores the Settings of the particular User on the Server. For example, every time you change some setting on the Web Client https://chat.susi.ai, an API call is made to aaa/changeUserSettings.json with appropriate parameters with information of the changed setting, and the User settings are stored to the Server in the Accounting Object of the User.

Implementation of a Servlet to store Device information on Server

The task of this servlet is to store the information of a new device (Smart speaker) whenever it is initially set up using the Android/iOS app.

This is the implementation of the 4 methods of a servlet which is used to store information of connected devices.

```
    @Override
    public String getAPIPath() {
        return "/aaa/addNewDevice.json";
    }

    @Override
    public UserRole getMinimalUserRole() {
        return UserRole.USER;
    }

    @Override
    public JSONObject getDefaultPermissions(UserRole baseUserRole) {
        return null;
    }

    @Override
   public ServiceResponse serviceImpl(Query query, HttpServletResponse response,
Authorization   authorization,   JsonObjectWithDefault   permissions)   throws
APIException {

               JSONObject value = new JSONObject();

               String key = query.get("macid", null);
               String name = query.get("name", null);
               String device = query.get("device", null);
               if (key == null || name == null || device == null) {
                        throw new APIException(400, "Bad service call, missing
arguments");
               } else {
                   value.put(name, device);
               }

           if (authorization.getIdentity() == null) {
                   throw new APIException(400, "Specified user data not found,
ensure you are logged in");
           }
           else {
                                                   Accounting   accounting   =
DAO.getAccounting(authorization.getIdentity());
                if (accounting.getJSON().has("devices")) {
                          accounting.getJSON().getJSONObject("devices").put(key,
value);
                    }
                else {
                       JSONObject jsonObject = new JSONObject();
                       jsonObject.put(key, value);
                       accounting.getJSON().put("devices", jsonObject);
                    }

               JSONObject result = new JSONObject(true);
               result.put("accepted", true);
               result.put("message", "You have successfully added the device!");
               return new ServiceResponse(result);
           }
    }
```

As it can be seen from the above code, the endpoint for this servlet is */aaa/addNewDevice.json* and it accepts 3 parameters -

- *macid* : Mac Address of the device
- *name* : Name of the device
- *device* : Additional information which you want to send about the device (subject to changes)

As the main task of this servlet is user specific, and should only be accessible to the particular user, hence we returned UserRole as USER in the getMinimalUserRole() method.

In the serviceImpl() method, first we extract the value of the URL query parameters and store them in variables. If any of the parameters is missing, we display an error response code of 400 with error message "Bad service call, missing arguments". If query parameters are fine, we store the *name* and *device* values in a new JSONObject, *value* in this case.

An if-else statement then checks for whether the User is logged in or not, using the authorization.getIdentity(), which is a function which returns the identity of the User. The implementation of this function is in Accounting.java file, and is as follows :

```
    public ClientIdentity getIdentity() {
        return identity;
    }
```

If the User is logged in, getAccounting() function of DAO.java file is called which returns an Accounting object according to the following implementation of the function :

```
    public static Accounting getAccounting(@Nonnull ClientIdentity identity) {
        return new Accounting(identity, accounting);
    }
```

Our device information is then stored in this Accounting object in the following format :

```
    {
  "lastLoginIP": "162.158.166.19",
  "accepted": true,
  "message": "Success: Showing user data",
  "session": {"identity": {
    "type": "email",
    "name": "myemail@gmail.com",
    "anonymous": false
  }},
  "settings": {
    "customThemeValue": "4285f4,f5f4f6,fff,f5f4f6,fff,4285f4,",
    "theme": "light"
  },
  "devices": {
    "MacID2": {"MyDevice": "SmartSpeaker"},
    "MacID1": {"Name of device": "SmartSpeaker"}
  }
}
```

Resources

- Files -
  - DAO.java
  - Accounting.java
  - ChangeUserSettings.java
  - ListUserSettings.java
- Blogs -
  - https://blog.fossasia.org/tag/susi-accounting/
- DAO -
  https://stackoverflow.com/questions/19154202/data-access-object-dao-in-java