SHM Web Interface

# login.jsp

TextField\_1 : username  
TextField\_2 : password  
Button\_1 : Login  
Button\_2 : Register

Button\_1\_action : validate user, get user privilige, onSuccess : main.jsp

@Controller = UserProfileController method = getUserProfile()

Button\_2\_action : registration.jsp

# registration.jsp

TextField\_1 : user\_id (required)  
TextField\_2 : user\_password (required)  
TextField\_3 : email (required)  
TextField\_4 : phone\_number (required)  
TextField\_5 : address  
TextField\_6 : city

Button\_1 : Register  
Button\_2 : Cancel

Button\_1\_action : validateTextFields, validateUserExistence, onSuccess : login.jsp

@Controller = UserProfileController method = addUserProfile() (if null -> userExists -> error)

Button\_2\_action : login.jsp

# main.jsp

The main page containing all tabs as:

1. productcontrol.jsp,
2. productregistration.jsp,
3. productsettings.jsp
4. adminpage.jsp

Landing page is productcontrol.jsp in case customer have registered product(s), else productregistration.jsp.

# productregistration.jsp

TextFiled\_1 : accept serial\_number

Button\_1 : Register

Button\_1\_action: submits serial\_number using

@Controller UserProductController method: registerProduct(userid : String, serial : Serial)

View\_1 : Show status of registration

Table\_1 : Show product related information

Button\_2 : Product Settings

Button\_2\_action : navigates to productsettings.jsp tab

# productsettings.jsp

TextView\_1 : MyProducts

List\_1 show user products, (serial\_number, product\_name) first product is the default to show

List\_1: populated from

@UserProductController method = getProductByUserId(userId : String) : List<UserProduct>

onDropDownItemSelected: gets UserProduct by serialNumber and userid, populates **UserProductTable RelaySettingTable** **and InputSettingTable**

@UserProductController method = getProductByUserId(userId : String, serialNumber : String) : UserProduct

**UserProductTable** : Provides data of UserProduct

Dropdown\_2 : user can select product settings here

UserProduct attributes :

* serial\_number (fix)
* name (editable)
* relay\_count (fix)
* input\_count (fix)
* phone\_number
* ssid
* password (for router)
* host
* port
* apn (as Service Provider aka Telenor/Vodafone) editable too
* wifi (has wifi?)
* gprs (has gprs?)
* gsm (has gsm?)

**RelaySettingTable** : Provides rows of RelaySetting *(relay\_count number of rows.)*

RelaySetting Attributes:

* relay\_id (fix) others editable
* relay\_name
* start\_timer
* end\_timer
* delay
* timer\_enabled
* delay\_enabled

onRelaySettingLineSelected : populates **UserControlSettingTable**

**UserControlSettingTable:**

UserControlSetting Attributes:

* user\_name (all editable)
* relay\_id
* sms\_access
* call\_access
* BUTTON(REMOVE), BUTTON(MODIFY), BUTTON(ADD)

All buttons use:

**InputSettingTalbe** (provides data of InputSettings, input\_count number of rows)

InputSetting Attributes

* input\_id (fix) others editable
* input\_name
* start\_timer
* end\_timer
* timer\_enabled
* value\_post\_fix
* sample\_rate

onInputSettingLineSelected: Populates UserProductTriggerSettingTable

**UserProductTriggerSettingTable :** populates data of UserProductSetting

**UserProductTriggerSetting** Attributes:

* input\_id
* trigger\_id
* trigger\_relay\_id (shows relay\_name)
* trigger\_enabled
* trigger\_value
* trigger\_state
* trigger\_action (button, onClick: alertBox) ez lényegtelen még
* BUTTON(REMOVE), BUTTON(MODIFY), BUTTON(ADD)

All buttons use:

Page Bottom:

BUTTON(SAVE) : saves all modified data with @Controller UserProductController method = updateUserProduct(userProduct : UserProduct)

BUTTON(CANCEL) : reverts data

BUTTON(ADD) : add data as a new setting

# productcontrol.jsp

Controls User Product remotely

TextField\_1 : displays number of connected clients with:

@Controller UserProductController method : getConnectedProducts(userid : String) : List<DeviceSession>

DropDown\_1 : displays all registered UserProducts, marks online ones with green circle

onDropDownItemSelected : populated RelayControlTable, InputOverviewTable

**RelayControlTable** : provides data of RelayControlSettings relay\_count number of rows

* state (ON/OFF)
* relay\_id
* relay\_name
* start\_timer (if enabled)
* end\_timer (if enabled)
* delay (if enabled)
* BUTTON(ON/OFF) ( if have access)

onButtonClicked :

@Controller UserProductController method = switchRelay(userid : String, serialNumber : String, relay : Integer, state : String) : String (state)

InputOverviewTable

InputOverview attributes :

* input\_id
* input\_name
* start\_timer (if enabled)
* end\_timer (if enabled)
* sample\_rate
* value + value\_post\_fix

value comes from

@Controller UserProductController method = getMeasuredValue(serialNumber : String, input\_id) : String

# adminpage.jsp

only seen by ADMINs

DOROPDOWN(PRODUCT\_CATEGORIES)

populated with product\_ids by using

@ProductCategoryController method = getProductCategories() : List<ProductCategory>

onDropDownItemSelected : populates **networkSettingsTable, relaySettingsTable, inputSettingsTable**

(same shit as for productsettings.jsp)

BUTTON(SAVE) – collects product information from Tables and submits new ProductCategory by using

@ProductCategoryController – method saveProductCategory(productCategory ProductCategory) : ProductCategory

in case product\_id exists, updates ProductCategory

**View** of registered products sortable by isRegistered, isActivated

Admin can register/unregister RegisteredProducts with

@ProductRegistrationController method = updateRegisteredProduct(registeredProduct : RegisteredProduct) : RegisteredProduct

**View** of UserProfiles, their products and like everything

Access for userLogs