

## EGI Vtiger CRM

---

### Contents

1. [Vtiger Customisations](#)
  1. [Google Analytics](#)
  2. [Enable DEBUG](#)
  3. [LDAP Authentication](#)
    1. [EGI SSO](#)
    2. [Debugging ldap queries](#)
    3. [How to implement LDAP user authentication in version 5.2.1](#)
    4. [How to implement LDAP user authentication in version 5.3.0](#)
    5. [How to implement LDAP user authentication in version 5.4.0](#)
  4. [Dashboards](#)
    1. [How to create a new dashboard](#)
    2. [New Dashboards implemented](#)
      1. [Query for top 10 countries](#)
      2. [Query for top 10 Scientific Disciplines](#)
    3. [Change Startup of Dashboard Home](#)
  5. [Workflows](#)
    1. [Code use cases](#)
    2. [References](#)
    3. [Project List](#)
  6. [Web Form Validation](#)
    1. [Define AjaxConditionalValidation](#)
    2. [Change CreateView.tpl](#)
    3. [Define AjaxProjectListCheck](#)
    4. [Execution Workflow](#)
  7. [Metrics Scripts](#)
  8. [SMARTY TEMPLATES](#)
    1. [Remove links form main menu](#)
  9. [Display changes](#)
    1. [How to deactivate module fields](#)
    2. [How to allow assignments to individual users](#)
    3. [Change Main Toolbar names \(changed after 540\)](#)
    4. [Change Tab Menus](#)
    5. [Change Home page Components values](#)
    6. [Changing an existing field: type pick-list](#)
    7. [How to remove the Java script edit per field](#)
  10. [Authorization Policies](#)

1. [Roles, Profiles and Users](#)
  2. [Roles, Groups and Users](#)
  3. [User based ACLs](#)
11. [Administrative tasks \(user:admin\)](#)
  1. [Home page Components](#)
12. [Disable Email](#)
2. [Development](#)
  1. [Implemented Debug Output to Screen Functions](#)
3. [Operation](#)
  1. [MySQL Backups](#)
    1. [VT.MY.DOMAIN](#)
    2. [VTDEV.MY.DOMAIN](#)
  2. [Servers Management](#)
  3. [HTTPD Configuration \(in production\)](#)
  4. [Utilities](#)
    1. [mysqlvtigercrm](#)
    2. [vtigerepos ""\(Not finished\)""](#)
    3. [How to enable Table Relation View in PHPmyadmin](#)
4. [CheckList for Upgrading](#)

# Vtiger Customisations

## Google Analytics

Insert the following piece of code before closing the **</head>** tag in Smarty/templates/Header.tpl

```
<head>
(...)
<script src="http://www.google-analytics.com/urchin.js" type="text/javascript">
</script>
<script type="text/javascript">
_uacct = "UA-30732656-1";
urchinTracker();
</script>
</head>
```

## Enable DEBUG

Activate vtiger debug mode (/var/www/html/vtigercrm/logs/vtigercrm.log)

```
# grep DEBUG `pwd`/config.performance.php  
'LOG4PHP_DEBUG' => true,
```

Change log4php.properties so that it contains:

```
log4php.rootLogger=DEBUG,A1  
log4php.appender.A1=LoggerAppenderRollingFile  
log4php.appender.A1.MaxFileSize=3MB  
log4php.appender.A1.MaxBackupIndex=10  
log4php.appender.A1.layout=LoggerLayoutTTCC  
log4php.appender.A1.layout.ContextPrinting="true"  
log4php.appender.A1.layout.DateFormat="%c"  
log4php.appender.A1.File=logs/vtigercrm.log
```

Add Debug to specific Functions

```
Just add the following line to the beginning of the function:  
Ex: function Name() {  
    global $adb; //Define $adb as a global variable  
    $adb->setDebug(true);  
}
```

## LDAP Authentication

### EGI SSO

- SSO service URI: ldaps://aldor.ics.muni.cz
- Basename for users: ou=People,dc=egi,dc=eu
- User DN example: uid=user045,ou=People,dc=egi,dc=eu

Connecting via LDAPS requires the validation of the remote server certificate which means that the whole trust chain must be verified, hence the CA certificates need to be installed. First configure ldap to search for CA root certs.

```
$ vi /etc/openldap/ldap.conf
```

```
TLS_CACERTDIR /etc/openldap/cacerts
```

Download the CA certs and install them in /etc/openldap/cacerts In this case the certificate for aldor.ics.muni.cz is signed by TERENA which uses the **AddTrust** commercial provider. The trust chain is as follows:

Terena Server Certificate CA Certificate chain:

```
+ AddTrust External CA Root
+ UTN-USERFirst-Hardware
+ TERENA SSL CA
```

The root certificates can be downloaded from TACAR at: <http://www.terena.org/activities/tcs/repository/> Download the three certificates in PEM format and copy them to /etc/openssl/cacerts then create the hash links for each one of the files by repeating the following command and replacing ca\_root\_file.pem by the appropriate file name.

```
ln -s ca_root_file.pem `openssl x509 -hash -noout -in ca_root_file.pem`.0
```

Done just do an ldapsearch to check it:

```
ldapsearch -x -D 'uid=jorge,ou=People,dc=egi,dc=eu' -H ldaps://aldor.ics.muni.cz:636 -W -b 'uid=jorge,ou=People,dc=egi,dc=eu'
```

## Debugging ldap queries

To check is ldap port is reachable the resulting port state must be **open**

```
nmap -P0 -p636 aldor.ics.muni.cz
```

To debug an ldap query with full logging add -d 65535

```
ldapsearch -d 65535 -x -D 'uid=jorge,ou=People,dc=egi,dc=eu' -H ldaps://aldor.ics.muni.cz:636 -W -b 'uid=jorge,ou=People,dc=egi,dc=eu'
```

To debug TLS / SSL issues (in this case LDAPS). This allows to obtain the remote server public key and to see other authentication details such as the certificate issuers, trust chain, etc

```
openssl s_client -host aldor.ics.muni.cz -port 636
```

## How to implement LDAP user authentication in version 5.2.1

### 0. LDAP server:

- Test ldap server in vtdev

Slapd Configuration file:

```
/etc/openldap/slap.conf
```

- read guidelines inside this wiki in [LDAP\\_authentication](#)
- example of created tree:

```
dn: uid=nabo,ou=People,dc=localhost,dc=localdomain
uid: nabo
cn: nabo
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
userPassword:: ****
shadowLastChange: 15351
shadowMin: 0
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/bash
uidNumber: 500
gidNumber: 500
homeDirectory: /home/nabo
structuralObjectClass: account
entryUUID: XXXXXX
creatorsName: cn=admin,dc=localhost,dc=localdomain
createTimestamp: 20120112142655Z
entryCSN: 20120112142655.031130Z#000000#000#000000
modifiersName: cn=admin,dc=localhost,dc=localdomain
modifyTimestamp: 20120112142655Z
```

1. Install and copy the files from the the LDAP patch called vtiger521-ldap-integration-b.zip[[<http://forums.vtiger.com/download/file.php?id=4830>]

- files modified in modules:

```
modules/Users/QueryLdap.php
modules/Users/EditView.php
modules/Users/Authenticate.php
modules/Users/language/en_us.lang.php
modules/Users/DetailView.php
modules/Users/Users.php
```

- files modified in Smarty:

```
Smarty/templates/DetailViewHidden.tpl
Smarty/templates/UserEditView.tpl
```

- new ldap include module:

```
include/ldap/adLdap.php
include/ldap/config.ldap.php
include/ldap/Ldap.php
```

2. Change and adapt vtiger **LDAP configuration file** ("include/ldap/config.ldap.php") There are two types of access configurations, the Local Host and the External Host.

```
// Login Type may be: 'LDAP' or 'AD' or 'SQL'
// Use 'SQL' to login using the passwords stored in the vTiger Sql database
$AUTHCFG['authType']      = 'LDAP';
```

### For Localhost Access

```
// ----- Configuration LDAP -----
$AUTHCFG['ldap_host']      = 'localhost';           //system where ldap is running (e.g. ldap://localhost)
$AUTHCFG['ldap_port']      = '389';                 //port of the ldap service
```

```
// The LDAP branch which stores the User Information
// This branch may have subfolders. PHP will search in all subfolders.
$AUTHCFG['ldap_basedn']    = 'ou=People,dc=localhost,dc=localdomain';

// The account on the LDAP server which has permissions to read the branch specified in ldap_basedn
//$AUTHCFG['ldap_username'] = 'cn=admin,dc=localhost,dc=localdomain'; // set = NULL if not required
$AUTHCFG['ldap_username'] = NULL; // set = NULL if not required
//$AUTHCFG['ldap_pass']     = 'admin'; // set = NULL if not required
$AUTHCFG['ldap_pass']     = NULL; // set = NULL if not required
```

```
// LOCALHOST Configuration
// Predefined LDAP fields (these settings work on Win 2003 Domain Controller)
$AUTHCFG['ldap_objclass']  = 'objectClass';
$AUTHCFG['ldap_account']   = 'cn';
$AUTHCFG['ldap_forename']  = 'givenName';
$AUTHCFG['ldap_lastname']  = 'sn';
$AUTHCFG['ldap_fullname']  = 'cn';
$AUTHCFG['ldap_email']     = 'mail';
$AUTHCFG['ldap_tel_work']  = 'telephoneNumber';
```

```
$AUTHCFG['ldap_department'] = 'physicalDeliveryOfficeName';
$AUTHCFG['ldap_description'] = 'description';
$AUTHCFG['sql_accounts']     = array("admin");           //the users whose authentication will be from database
instead of from ldap
```

```
// Required to search users: the array defined in ldap_objclass must contain at least one of the following
values
$AUTHCFG['ldap_userfilter'] = 'user|person|organizationalPerson|account';
// LOCALHOST Configuration
```

## For External Access

```
// ----- Configuration Secure LDAP / External Source -----
$AUTHCFG['ldap_host']      = 'ldaps://aldor.ics.muni.cz';           //system where ldap is running (e.g.
ldaps://domainname or ipaddress)
$AUTHCFG['ldap_port']      = '636';                               //port of the ldaps service
```

```
$AUTHCFG['ldap_basedn']    = 'ou=People,dc=egi,dc=eu';
$AUTHCFG['ldap_username']  = 'cn=crm,dc=egi,dc=eu';
$AUTHCFG['ldap_pass']      = 'password';
```

```
// ----- Predefined LDAP fields (these settings work on Win 2003 Domain Controller) -----
$AUTHCFG['ldap_objclass']  = 'dn';
$AUTHCFG['ldap_account']   = 'uid';
$AUTHCFG['ldap_forename']  = 'givenName';
$AUTHCFG['ldap_lastname']  = 'sn';
$AUTHCFG['ldap_fullname']  = 'uid';
$AUTHCFG['ldap_email']     = 'mail';
$AUTHCFG['ldap_tel_work']  = 'telephoneNumber';
$AUTHCFG['ldap_department'] = 'physicalDeliveryOfficeName';
$AUTHCFG['ldap_description'] = 'description';
```

```
$AUTHCFG['sql_accounts']   = array("admin");           //the users whose authentication will be from database
instead of from ldap
```

```
// Required to search users: the array defined in ldap_objclass must contain at least one of the following
values
$AUTHCFG['ldap_userfilter'] = 'uid|cn|sn';
```

## COMMON

```
//Users can change their own password or not (new add not in default 5.3.0 installation)
// $AUTHCFG['ldap_allowpasswordchange'] = true;
// $AUTHCFG['ldap_admin'] = 'cn=admin,dc=example,dc=com'; //Change to NULL if admin do not need to change own
password through vTiger
// $AUTHCFG['ldap_adminpwd'] = 'mysecret'; //Change to NULL if admin do not need to change own password through
vTiger
$AUTHCFG['ldap_allowpasswordchange'] = false;
$AUTHCFG['ldap_admin'] = NULL; //Change to NULL if admin do not need to change own password through vTiger
$AUTHCFG['ldap_adminpwd'] = NULL; //Change to NULL if admin do not need to change own password through vTiger
// END
```

## How to implement LDAP user authentication in version 5.3.0

1. Use 5.2.1 workaround in section **How\_to\_implement\_LDAP\_user\_authentication\_in\_version\_5.2.1** but **don't overwrite "modules/Users/Users.php"**

2. **Code changed in module Users** (default 5.3.0 installation):

- new include added (beginning of file)

```
require_once 'include/ldap/config.ldap.php';
require_once 'include/ldap/Ldap.php';
```

- function doLogin

```
function doLogin($user_password) {
    global $AUTHCFG;
    $usr_name = $this->column_fields["user_name"];
    //PINA / CARLOS
    // Allow the 'admin' always to log in independent from the LDAP server
    $usr_list = $AUTHCFG['sql_accounts'];

    if (in_array(strtolower($usr_name), $usr_list))
        $AUTHCFG['authType'] = 'SQL';

    //SEAN TSANG PATCH START - User must exists in database first
    $query = "SELECT * from $this->table_name where user_name=?";
    $result = $this->db->requirePsSingleResult($query, array($usr_name), false);
    if (empty($result)) return false;
    //SEAN TSANG PATCH END-----
    //PINA / CARLOS
```



```

switch (strtoupper($AUTHCFG['authType'])) {
    case 'LDAP':
        $this->log->debug("Using LDAP authentication");
        // require_once('modules/Users/authTypes/LDAP.php');
        require_once('include/ldap/Ldap.php');
        $result = ldapAuthenticate($this->column_fields["user_name"],
$user_password);

        if ($result == NULL) {
            return false;
        } else {
            return true;
        }
        break;

    case 'AD':
        $this->log->debug("Using Active Directory authentication");
        require_once('modules/Users/authTypes/adLDAP.php');
        $adldap = new adLDAP();
        if ($adldap->authenticate($this->column_fields["user_name"], $user_password))
{
            return true;
        } else {
            return false;
        }
        break;

    default:
        $this->log->debug("Using integrated/SQL authentication");
        $query = "SELECT crypt_type FROM $this->table_name WHERE user_name=?";
        $result = $this->db->requirePsSingleResult($query, array($usr_name), false);
        if (empty($result)) {
            return false;
        }
        $crypt_type = $this->db->query_result($result, 0, 'crypt_type');
        $encrypted_password = $this->encrypt_password($user_password, $crypt_type);
        $query = "SELECT * from $this->table_name where user_name=? AND
user_password=?";
        $result = $this->db->requirePsSingleResult($query, array($usr_name,
$encrypted_password), false);
        if (empty($result)) {
            return false;
        } else {
            return true;
        }
}

```

```

        }
        break;
    }
    return false;
}

```

**3. For External Server Access code changed in includes/ldap/Ldap.php** (default 5.3.0 installation statements commented as #ORIGINAL):

- function ldapSearchUserAccountAndName

```

// Searches for a user's fullname
// returns a hashtable with Account => FullName of all matching users
function ldapSearchUserAccountAndName($user){
    global $AUTHCFG;
    $fldaccount = strtolower($AUTHCFG['ldap_account']); //uid
    $fldname    = strtolower($AUTHCFG['ldap_fullname']); //uid
    $fldclass   = strtolower($AUTHCFG['ldap_objclass']); //dn
    # ORIGINAL $usrfilter = explode("|", $AUTHCFG['ldap_userfilter']);
    $usrfilter = explode("|", $AUTHCFG['ldap_userfilter']); // "uid","cn","sn"
    $required   = array($fldaccount,$fldname,$fldclass);
    // ORIGINAL $ldapArray = ldapSearchUser("$fldname=$user*", $required);
    $ldapArray = ldapSearchUser("$fldname=$user*", $required);
    // copy from LDAP specific array to a standardized hashtable
    // Skip Groups and Organizational Units. Copy only users.
    for ($i=0; $i<$ldapArray["count"]; $i++)
    {
        $isuser = false;
        foreach($usrfilter as $filt)
        {
            // ORIGINAL if (in_array($filt, $ldapArray[$i][$fldclass]))
            if (in_array($filt, $ldapArray[$i]))
            {
                $isuser = true;
                break;
            }
        }
        if ($isuser)
        {
            $account = $ldapArray[$i][$fldaccount][0];
            $name     = $ldapArray[$i][$fldname][0];
            $userArray[$account] = $name;
        }
    }
}

```

```

    }
    return $userArray;
}

```

- function ldapSearchUser

```

// Search a user by the given filter and returns the attributes defined in the array $required
function ldapSearchUser($filter, $required){
    global $AUTHCFG;
    $conn = ldapConnectServer();
    if ($conn == NULL)
        return NULL;
    $ident=@ldap_search($conn, $AUTHCFG['ldap_basedn'], $filter, $required);
    if ($ident)
    {
        $result = ldap_get_entries($conn, $ident);
        ldap_free_result($ident);
    }
    ldap_unbind($conn);
    return $result;
}

```

## How to implement LDAP user authentication in version 5.4.0

- function ldapSearchUser

```

* function ldapSearchUser($filter, $required){
    global $AUTHCFG;
    $conn = ldapConnectServer();
    if ($conn == NULL)
        return NULL;
    $ident=@ldap_search($conn, $AUTHCFG['ldap_basedn'], $filter, $required);
    if ($ident)
    {
        $result = ldap_get_entries($conn, $ident);
        ldap_free_result($ident);
    }
    ldap_unbind($conn);
    return $result;
}

```

- function ldapSearchUserAccountAndName

```

function ldapSearchUserAccountAndName($user){
    global $AUTHCFG;
    $fldaccount = strtolower($AUTHCFG['ldap_account']); //uid
    $fldname = strtolower($AUTHCFG['ldap_fullname']); //uid
    $fldclass = strtolower($AUTHCFG['ldap_objclass']); //dn
    # ORIGINAL $usrfilter = explode("|", $AUTHCFG['ldap_userfilter']);
    $usrfilter = explode("|", $AUTHCFG['ldap_userfilter']); // "uid","cn","sn"
    $required = array($fldaccount,$fldname,$fldclass);
    // ORIGINAL $ldapArray = ldapSearchUser("$fldname=$user*", $required);
    $ldapArray = ldapSearchUser("$fldname=$user*", $required);
    // copy from LDAP specific array to a standardized hashtable
    // Skip Groups and Organizational Units. Copy only users.
    for ($i=0; $i<$ldapArray["count"]; $i++)
    {
        $isuser = false;
        foreach($usrfilter as $filt)
        {
            // ORIGINAL if (in_array($filt, $ldapArray[$i][$fldclass]))
            if (in_array($filt, $ldapArray[$i]))
            {
                $isuser = true;
                break;
            }
        }
        if ($isuser)
        {
            $account = $ldapArray[$i][$fldaccount][0];
            $name = $ldapArray[$i][$fldname][0];
            $userArray[$account] = $name;
        }
    }
    return $userArray;
}

```

## Dashboards

### How to create a new dashboard

Example of how to implement a Dashboard for **Contacts by Account**

0. In "modules/Home/HomestuffAjax.php":

- Define "\$dashbd\_strings['contactbyaccount']" in the "\$graph\_array"

```
$graph_array = array("leadsource" => $dashbd_strings['leadsource'],  
    "leadstatus" => $dashbd_strings['leadstatus'],  
    "leadindustry" => $dashbd_strings['leadindustry'],  
    (...)  
    "contactbyaccount"=> $dashbd_strings['contactbyaccount'],  
    "contactbycampaign"=> $dashbd_strings['contactbycampaign'],  
    "ticketsbyaccount"=> $dashbd_strings['ticketsbyaccount'],  
    "ticketsbycontact"=> $dashbd_strings['ticketsbycontact'],  
    );
```

### 1. In "modules/Dashboard/index.php":

- Define "\$mod\_strings['contactbyaccount']" in the "\$graph\_array"
- It is here that you hide / show the available dashboards

```
$graph_array = Array(  
    "DashboardHome" => $mod_strings['DashboardHome'],  
    "leadsource" => $mod_strings['leadsource'],  
    "leadstatus" => $mod_strings['leadstatus'],  
    (...)  
    "contactbyaccount"=> $mod_strings['contactbyaccount'],  
    "contactbycampaign"=> $mod_strings['contactbycampaign'],  
    "ticketsbyaccount"=> $mod_strings['ticketsbyaccount'],  
    "ticketsbycontact"=> $mod_strings['ticketsbycontact'],  
    );
```

### 2. In "modules/Dashboard/homestuff.php":

- Define "\$mod\_strings['contactbyaccount']" in the "\$graph\_array"

```
$graph_array = Array(  
    "DashboardHome" => $mod_strings['DashboardHome'],  
    "leadsource" => $mod_strings['leadsource'],  
    "leadstatus" => $mod_strings['leadstatus'],  
    (...)  
    "contactbyaccount"=> $mod_strings['contactbyaccount'],  
    "contactbycampaign"=> $mod_strings['contactbycampaign'],
```

```
"ticketsbyaccount"=> $mod_strings['ticketsbyaccount'],
"ticketsbycontact"=> $mod_strings['ticketsbycontact'],
);
```

### 3. In "modules/Dashboard/language/en\_us.lang.php":

- Introduce the "contactbycampaign" string

```
"leadsource" => "Leads By Source",
"leadstatus" => "Leads By Status",
"leadindustry" => "Leads By Industry",
(...)
"contactbyaccount"=>"Contacts by Account",
"contactbycampaign"=>"Contacts by Campaign",
"ticketsbyaccount"=>"Tickets by Account",
"ticketsbycontact"=>"Tickets by Contact",
```

### 4. In "modules/Dashboard/display\_charts.php":

- Define the mysql query

```
//Query for contacts by account
$contacts_by_account="
select vtiger_contactdetails.*, vtiger_groups.groupname AS ticketgroupname, vtiger_crmentity.*,
vtiger_account.*
from vtiger_contactdetails
inner join vtiger_crmentity on vtiger_crmentity.crmid=vtiger_contactdetails.contactid
inner join vtiger_account on vtiger_account.accountid=vtiger_contactdetails.accountid
left join vtiger_groups on vtiger_groups.groupid=vtiger_crmentity.smownerid
left join vtiger_users on vtiger_crmentity.smownerid=vtiger_users.id
where vtiger_crmentity.deleted=0";
```

- Define "\$mod\_strings['contactbyaccount']" in the "\$graph\_array"

```
$graph_array = Array(
    "DashboardHome" => $mod_strings['DashboardHome'],
    "leadsource" => $mod_strings['leadsource'],
    "leadstatus" => $mod_strings['leadstatus'],
    (...)
```

```

        "contactbyaccount"=> $mod_strings['contactbyaccount'],
        "contactbycampaign"=> $mod_strings['contactbycampaign'],
        "ticketsbyaccount"=> $mod_strings['ticketsbyaccount'],
        "ticketsbycontact"=> $mod_strings['ticketsbycontact'],
    );

```

- Insert the code for the graph generation taking into account that the graph can

be generated in two different situations:

- In the homepage
- via the "Analytics" module

```

if(isset($_REQUEST['from_page']) && $_REQUEST['from_page'] == 'HomePage'){
    if( ... ) {
        (...)
    }
    //Contacts by Account
    elseif ($profileTabsPermission[getTabid("Accounts")] == 0
    && ($type == "contactbyaccount")
    && $profileTabsPermission[getTabid("Contacts")] == 0 ) {
        $graph_by="accountid";
        $graph_title=$mod_strings['contactbyaccount'];
        $module="Accounts";
        $where="";
        $query=getDashboardQuery($contacts_by_account,$module);
        return get_graph_by_type($graph_by,$graph_title,$module,$where,$query,"210","210","forhomepage");
    }
}
else {
    if( ... ) {
        (...)
    }
    //Contacts by Account
    elseif ($profileTabsPermission[getTabid("Accounts")] == 0
    && ($type == "contactbyaccount")
    && $profileTabsPermission[getTabid("Contacts")] == 0 ){
        $graph_by="accountid";
        $graph_title=$mod_strings['contactbyaccount'];
        $module="Accounts";
        $where="";
        $query=getDashboardQuery($contacts_by_account,$module);
        echo get_graph_by_type($graph_by,$graph_title,$module,$where,$query);
    }
}

```

```
}
}
```

## New Dashboards implemented

### Query for top 10 countries

- Change *vtigercrm/modules/Dashboard/language/en\_us.lang.php*

```
"accountbyproject" => "Account by Country (top 10)"
```

- Get the top 10 countries

```
$get_country = "select vtiger_accountscf.cf_635, count(1) AS number from vtiger_account
inner join vtiger_accountscf on vtiger_account.accountid = vtiger_accountscf.accountid
inner join vtiger_crmentity on vtiger_account.accountid=vtiger_crmentity.crmid
where vtiger_crmentity.deleted=0 group by vtiger_accountscf.cf_635 order by number DESC limit 10";
```

- Next we build the second query with all account belonging to the top 10 country

```
$count_C = getDashboardQuery($get_country,"Accounts");
$result = $adb->pquery($count_C, array());
$num_conts = $adb->num_rows($result);
$cont_Array = array();
if(count($num_conts) > 0){
    for($z=1;$z<$num_conts;$z++){
        $cont_ID=$adb->query_result($result,$z,'cf_635');
        $country_add = " OR vtiger_accountscf.cf_635='$cont_ID'".$country_add;
    }
    $cont_ID=$adb->query_result($result,0,'cf_635');
    $country_check ="vtiger_accountscf.cf_635='$cont_ID'".$country_add;
}
```

- Final query

```
$account_by_countrie="select cf_635 from vtiger_accountscf
inner join vtiger_crmentity on vtiger_crmentity.crmid=vtiger_accountscf.accountid
inner join vtiger_account on vtiger_account.accountid = vtiger_accountscf.accountid
where vtiger_crmentity.deleted=0 and ( ".$country_check." );
```



## Query for top 10 Scientific Disciplines

```
// query for top 10 disciplines
$get_disciplines = "select vtiger_accountscf.cf_667, count(1) AS number from vtiger_account inner join
vtiger_accountscf on
vtiger_account.accountid = vtiger_accountscf.accountid inner join vtiger_crmentity on
vtiger_account.accountid=vtiger_crmentity.crmid
where vtiger_crmentity.deleted=0 group by vtiger_accountscf.cf_667 order by number DESC limit 10";
```

- Next we build the second query with all account belonging to the top 10 scientific disciplines

```
$count_C = getDashboardQuery($get_disciplines,"Accounts");
$result = $adb->pquery($count_C, array());
$num_conts = $adb->num_rows($result);
$cont_Array = array();
    if(count($num_conts) > 0){
        for($z=1;$z<$num_conts;$z++){
            $cont_IX=$adb->query_result($result,$z,'cf_667');
            $discipline_add = " OR vtiger_accountscf.cf_667='$cont_IX'".$discipline_add;
        }
        $cont_IX=$adb->query_result($result,0,'cf_667');
        echo "Name: ".$cont_IX;
        $country_check = " where vtiger_accountscf.cf_635='$cont_ID'".$country_add;
        $discipline_check ="vtiger_accountscf.cf_667='$cont_IX'".$discipline_add;
    }
```

- Final query

```
$account_by_discipline ="select cf_667 from vtiger_accountscf inner join vtiger_crmentity on
vtiger_crmentity.crmid=vtiger_accountscf.accountid
inner join vtiger_account on vtiger_account.accountid = vtiger_accountscf.accountid where
vtiger_crmentity.deleted=0 and
(".$discipline_check.)";
```

- Last but not least implement it into dashboard:

```
//Charts for Account by discipline
elseif ($profileTabsPermission[getTabid("Accounts")] == 0 && ($type ==
"accountbydiscipline") && $profileTabsPermission[getTabid("Accounts")] == 0 )
{
    $graph_by="cf_667";
    $graph_title=$mod_strings['accountbydiscipline'];
```

```

        $module="Accounts";
        $where="";
        $query=getDashboardQuery($account_by_discipline,$module);
        return
    get_graph_by_type($graph_by,$graph_title,$module,$where,$query,"210","210","forhomepage");
}

```

## Change Startup of Dashboard Home

- To avoid the load of the graphs implemented in *DashboardHome.php* script, change the **modules/Dashboard/index.php** file as:

```

if(!isset($_REQUEST['type']))
{
    if(isset($_REQUEST['display_view']) && $_REQUEST['display_view'] == 'MATRIX')
    {
        require_once('modules/Dashboard/DashboardHome_matrix.php');
    }else
    {
        //
        require_once('modules/Dashboard/DashboardHome.php');
        require_once('modules/Dashboard/loadDashBoard.php');
    }
} else
{
    require_once('modules/Dashboard/loadDashBoard.php');
}/* Change Startup of Dashboard Home */

```

- loadDashBoard.php is called the first time with a generic URL and without passing through the **JavaScript** function **loaddashboard** present in the index.php. No graphs are constructed and an annoying initial message is displayed "Sorry no ...."
- To avoid the previous annoying message, change *modules/Dashboard/display\_charts.php*, and change the variable **LBL\_NO\_PERMISSION\_FIELD** to **LBL\_NO\_PERMISSION\_NEW** with empty content.
- To force loading the first value from "dashboard\_combo" automatically, add at the end of the *modules/Dashboard/index.php* file, the following code before the tag (`</script>`)

```

var grabEl = document.getElementById("dashboard_combo");
window.onload = loadDashBoard(grabEl);

```

## Workflows

This section presents the steps on how to defined custom scripts to be executed in workflows

- if the /modules/Workflow/ folder does not already exist, then create it.
- Copy the following file into it (/modules/Workflow/myExample.inc). It is just a simple function to send yourself an email, but you can put whatever code you want in this function:

```
<?php
/*
    Workflows Invoke Custom Function example Method Function file (myExample.inc)
    Put this file in the modules/Workflow/ directory. If the directory does not exist, then just create
it.
    See the myExample_REGISTER.php file for more information.
*/

function myExample($entity){

    //$entity is global object. Uncomment the code below to see everything it contains.
    // Ex: After adding a new contact, you can retrieve the Contact Number ID with:
    //     $contact_no = $entity->data[contact_no];
    //DEBUG ONLY - USE THIS TO DISPLAY $entity
    /*
        $entityArray = get_object_vars($entity);
        echo '< pre >';
        print_r($entityArray);
        echo '< /pre >';
    */

    //This example just sends you an email. Yes, I know that feature is
    // already available, but this is just an example to show you how
    // to set it all up!
    mail('me@mydomain.com','test subject','test message','From: me@mydomain.com');
}
?>
```

- Copy the file below (myExample\_REGISTER.php) into your vtiger root directory:

```
<?php
/*
    Workflows Invoke Custom Function example Register file (myExample_REGISTER.php)

    This will register a specific method function created in the myExample.inc
    This method function can then be invoked under:
    Workflows > New Condition > New Task > Invoke Custom Function

    HOW TO REGISTER FUNCTION:
```

In order to register the function, this Register file needs to be placed in your vTiger root directory, and then you manually need to call it from your browser: [http://www.mydomain.com/vtiger/myExample\\_REGISTER.php](http://www.mydomain.com/vtiger/myExample_REGISTER.php)

WARNING: You only want to call this register file ONCE...  
Everytime you call this register file, it will add the function to the vtiger\_workflowtasks\_entitymethod table in the database, so if you call it more than once then it will get added to the database more than once.

To REMOVE a registered method function, you need to manually delete it from the vtiger\_workflowtasks\_entitymethod table in the database.

```
$emm->addEntityMethod("ModuleName", "MyMethodName", "modules/Workflow/myExample.inc",
"myExampleFunctionName");
    ModuleName - Valid module names are: Contacts, PurchaseOrder, ???more???
    MyMethodName - This is the name that will appear in the drop down list for you to choose from.
    modules/Workflow/myExample.inc - The path and filename that contains the method function.
    If the modules/Workflow/ directory does not exist, then just create it!
    myMethodFunctionName - The name of the method function declared in myExample.inc
*/

require_once 'include/utils/utils.php';
require 'modules/com_vtiger_workflow/VTEntityMethodManager.inc';
$emm = new VTEntityMethodManager($adb);
$emm->addEntityMethod("Contacts", "myExample Send Me an Email", "modules/Workflow/myExample.inc",
"myExample");
echo 'addEntityMethod complete!';
?>
```

- Register your custom function in the database, open your browser and type the following in the address bar:

[https://www.mydomain.com/vtiger/myExample\\_REGISTER.php](https://www.mydomain.com/vtiger/myExample_REGISTER.php)

- Now that the function is registered in the database, you can either delete myExample\_REGISTER.php or you could instead move it to the /modules/Workflow folder if you want to keep it for future reference.
- Login into vTiger and (as an example)
  - Go to Settings > Workflows > New Workflow.
  - Choose "Contacts" then "Create".
  - Enter "Description", then choose "Only On The First Save", then "New Condition".
  - Select Assistant then 'does not contain' then 'bla bla bla' then click SAVE.
  - Then click New Task then Invoke Custom Function then Create.
  - Enter a Task Title and then choose **MyExample** from the Method Name drop down and then click SAVE.
- Congratulations, you will now receive an email everytime a new Contact is added!

## Code use cases

We have use the previous workflow implementation for the following use cases:

- Automatic update of project list: [vTiger\\_erm.tar.gz](#)

## References

- <http://forums.vtiger.com/viewtopic.php?t=31587>

## Project List

This form box was originally one, but now it's divided in 5 independent boxes:

- ESFRI Project
- ESFRI Cluster Project
- National Project
- Other International Project
- INFRA Project

## Web Form Validation

The validation of the **account** information is validated running javascript functions before submitting the form. The execution of the javascript functions are triggered clicked on the Save button of the account creation webpage. To change the validation mechanisms, we had to:

- Define a new javascript function, called *AjaxConditionalValidation*, to implement the validation mechanisms we are interested in,
- Change the *CreateView.tpl* so that *AjaxConditionalValidation* is called once the Save button is clicked

## Define AjaxConditionalValidation

```
# include/js/general.js
(...)

function AjaxConditionalValidation(module,fieldname1,fieldname2,fieldname3,fieldname4,oform)
{
    var fvalue1 = encodeURIComponent(trim(getObj(fieldname1).value));
    var fvalue2 = encodeURIComponent(trim(getObj(fieldname2).value));
    var fvalue3 = encodeURIComponent(trim(getObj(fieldname3).value));
    var fvalue4 = encodeURIComponent(trim(getObj(fieldname4).value));
    var recordid = getObj('record').value;
```

```
var brokenstring = fvalue3.split("%20",1);
var projectlist = brokenstring[0]+":"+fvalue2;

if( fvalue1 == ){
    alert('Account Name can not be empty');
    return false;
}
if( fvalue2 == && fvalue3 != 'Research%20Institute' ){
    alert('Account Accronym can not be empty for this type of Account');
    return false;
}

// Create the myregexp and use the method test
// The encodeURIComponent function has encoded white spaces
// and we had to go back to the source trim(getObj(fieldname2).value)
var myregexp = /^\\S+\\s+.*$/g.test(trim(getObj(fieldname2).value));
if ( myregexp ){
    alert('Account Accronym can not contain empty spaces');
    return false;
}

if( fvalue3 == ){
    alert('Account Type can not be empty');
    return false;
}

if( (fvalue4 == '--None--' || fvalue4 == ) && fvalue3 == 'ESFRI%20Project' ){
    alert('Scientific Discipline cannot be empty for ESFRI accounts');
    return false;
}
if( (fvalue4 == '--None--' || fvalue4 == ) && fvalue3 == 'INFRA%20Project' ){
    alert('Scientific Discipline cannot be empty for INFRA accounts');
    return false;
}

return true;
}
```

[Change CreateView.tpl](#)

Note: Please take in consideration that Smarty templates use "{" and "}" as special chars. To escape those chars, "{" should be

represented by "{ldelim}" and "}" by "{rdelim}"

```
$ vim Smarty/templates/CreateView.tpl
(...)
<input title="{ $APP.LBL_SAVE_BUTTON_TITLE}" accessKey="{ $APP.LBL_SAVE_BUTTON_KEY}" class="crmbutton small save"
onclick="this.form.action.value='Save';
if(AjaxConditionalValidation('Accounts','accountname','cf_637','cf_636',this.form)){ldelim}if(formValidate())
{ldelim}AjaxDuplicateValidate('Accounts','accountname',this.form);{rdelim}{rdelim}"
type="button" name="button" value=" { $APP.LBL_SAVE_BUTTON_LABEL} " style="width:70px" >
```

## Define AjaxProjectListCheck

- This also calls modules/Accounts/Save.php function which was altered

// Project List must be unique when account creation

### 1. include/js/general.js

```
function AjaxProjectListCheck(module,fieldname1,fieldname2,fieldname3,oform)
{
    var fieldvalue1 = encodeURIComponent(trim(getObj(fieldname1).value));
    var fieldvalue2 = encodeURIComponent(trim(getObj(fieldname2).value));
    var fieldvalue3 = encodeURIComponent(trim(getObj(fieldname3).value));
    var recordid = getObj('record').value;
    var brokenstring = fieldvalue3.split("%20",1);
    var projectlist = brokenstring[0]+":"+fieldvalue2;
    VtigerJS_DialogBox.block();
    var url = "module="+module+"&action="+module+"Ajax&file=Save&"+fieldname1+"="+fieldvalue1+"
    &pina_check=true&record="+recordid+"&projectlist="+projectlist;
    new Ajax.Request(
        'index.php',
        {
            queue: {
                position: 'end',
                scope: 'command'
            },
            method: 'post',
            postBody:url,
            onComplete: function(response) {
                var str = response.responseText
                if(str.indexOf('SUCCESS') > -1)
                {
```

```

                                oform.submit();
                                }else
                                {
                                    VtigerJS_DialogBox.unblock();
                                    alert(str);
                                    return false;
                                }
                            }
                        }
                    }
                );
            }
        }
    }
}

```

## 2. module/Accounts/Save.php

```

$search=vtlib_purify($_REQUEST['search_url']);
if(isset($_REQUEST['pina_check']) && $_REQUEST['pina_check'] != )
{
    //started
    $value = $_REQUEST['projectlist'];
    $params = array($value);
    $query = "select cf_626 from vtiger_cf_626 where cf_626=? " ;
    $result = $adb->pquery($query, $params);
    $value1 = $_REQUEST['accountname'];
    $query1 = "SELECT accountname FROM vtiger_account,vtiger_crmentity WHERE accountname =?
and vtiger_account.accountid = vtiger_crmentity.crmid and vtiger_crmentity.deleted != 1";
    $params1 = array($value1);
    $result1 = $adb->pquery($query1, $params1);
    if($adb->num_rows($result) > 0)
    {
        echo $mod_strings['LBL_PROJECTLIST_EXIST'];
    }
    elseif($adb->num_rows($result1) > 0)
    {
        echo $mod_strings['LBL_ACCOUNT_EXIST'];
    }
    else
    {
        echo 'SUCCESS';
    }
    die;
}

```



### 3. Smarty/templates/CreateView.tpl

- For 530 version

```
if(AjaxConditionalValidation('Accounts','accountname','cf_637','cf_636','industry',this.form))
{ldelim}if(formValidate())(AjaxProjectListCheck('Accounts','accountname','cf_637','cf_636',this.form));
{rdelim}
```

- For 540 version

```
if(AjaxConditionalValidation('Accounts','accountname','cf_637','cf_636','cf_667',this.form))
if(formValidate()) {ldelim}
if(AjaxDuplicateValidate('Accounts','accountname',this.form)) {ldelim} AddressSync(this.form,{ $ID});
{rdelim} {rdelim}"
type="button" name="button" value=" { $APP.LBL_SAVE_BUTTON_LABEL} " style="width:70px" >
```

### 4. Smarty/templates/salesEditView.tpl

Apply same changes as in 3 (this template is used when editing)

#### Execution Workflow

- When the **Save** button is clicked, it triggers the execution of the *AjaxConditionalValidation* javascript function.
- *AjaxConditionalValidation* receives as input the following account field names
  - *accountname* stands for the account name, and can not be empty
  - *cf\_636* stands for the account type and can not be empty
  - *cf\_637* stands for the account acronym and can not be empty if the account type is different than "Research Institution"
- If all the validations are correct, the *AjaxConditionalValidation* javascript function returns true, and the *AjaxDuplicateValidate* function is called to check if there are duplicated entries in the database for the same account name.
- If everything is ok, the form is submitted by *AjaxConditionalValidation*, and the modules/Account/Save.php file is called.

#### Metrics Scripts

```
[root@vt vtigercrm]# ll /var/www/html/vtigercrm/*metrics*
-rwxr-xr-x 1 root root 15122 Oct 19 10:16 metrics_report_accounts.php
-rwxr-xr-x 1 root root 11542 Oct 19 10:16 metrics_report_accounts.php.saved
-rwxr-xr-x 1 root root 5887 Oct 19 10:16 metrics_report_contacts.php
-rwxr-xr-x 1 root root 5701 Oct 19 10:16 metrics_report_documents.php
-rw-r--r-- 1 root root 7091 Oct 19 10:16 metrics_report_index_functions.php
-rw-r--r-- 1 root root 2407 Oct 19 10:16 metrics_report_index.php
```

## SMARTY TEMPLATES

- Location

/Smarty/templates/

### Remove links form main menu

Change "/Smarty/templates/Header.tpl"

- Gmail Bookmarklet (line 123):

```
<td style="padding-left:10px;padding-right:10px" class=small nowrap>
  {$GMAIL_BOOKMARKLET}
</td>
```

## Display changes

### How to deactivate module fields

These instructions hide deactivate some preconfigured fields to users with any kind of roles.

1. Login as Administrator
2. Select *CRM Settings* (top-right menu)
3. Select *Module Manager*
4. Select *Organizations / Contacts / Documents* (Hammer button)
5. Select *Layout Editor*
6. Edit the field you want to deactivate
7. Uncheck the *active* checkbox
8. To check the hidden / deactivated objects, select the *Hidden Fields* icon on the top-left on each block

### How to allow assignments to individual users

The default vtiger behaviour is not to allow assignments of **ToDo's**, Accounts or Contacts to groups or roles under another hierarchy branch.

So that all users are displayed in the assignment fields of modules, the following code has to be changed (vtiger 5.4.0)

```
$ cat -n /var/www/html/vtigercrm/include/utils/CommonUtils.php
(...)
```

```

1878          // Start change by LIP - 11/12/2012
1879          // Comment instructions to allow displaying all users in the user combo box
1880          // if ($is_admin == false && $profileGlobalPermission[2] == 1 &&
($defaultOrgSharingPermission[getTabid($module)] == 3 or $defaultOrgSharingPermission[getTabid($module)] == 0)) {
1881          //          $users_combo = get_select_options_array(get_user_array(FALSE, "Active", $current_user->id,
'private'), $current_user->id);
1882          //} else {
1883          $users_combo = get_select_options_array(get_user_array(FALSE, "Active", $current_user->id), $current_user->id);
1884          //}
1885          // End change by LIP - 11/12/2012
(...)

```

```

$ cat -n /var/www/html/vtigercrm/include/utils/EditViewUtils.php
(...)
290          // Start change by LIP - 11/12/2012
291          //Control will come here only for Products - Handler and Quotes - Inventory Manager
292          // if($is_admin==false && $profileGlobalPermission[2] == 1 &&
($defaultOrgSharingPermission[getTabid($module_name)] == 3 or
$defaultOrgSharingPermission[getTabid($module_name)] == 0))
293          // {
294          //          $users_combo = get_select_options_array(get_user_array(FALSE, "Active",
$assigned_user_id,'private'), $assigned_user_id);
295          // }
296          // else
297          // {
298          $users_combo = get_select_options_array(get_user_array(FALSE, "Active",
$assigned_user_id), $assigned_user_id);
299          //}
300          // End change by LIP - 11/12/2012
(...)
333          // Start change by LIP - 11/12/2012
334          // if($fieldname == 'assigned_user_id' && $is_admin==false && $profileGlobalPermission[2]
== 1 && ($defaultOrgSharingPermission[getTabid($module_name)] == 3 or
$defaultOrgSharingPermission[getTabid($module_name)] == 0))
335          // {
336          //          $users_combo = get_select_options_array(get_user_array(FALSE, "Active",
$assigned_user_id,'private'), $assigned_user_id);
337          // }
338          // else
339          // {
340          $users_combo = get_select_options_array(get_user_array(FALSE, "Active",

```

```
$assigned_user_id), $assigned_user_id);
341          // }
342          // End change by LIP - 11/12/2012
(...)
```

## Change Main Toolbar names (changed after 540)

vtiger home directory, open the "include/language/en\_us.lang.php"

## Change Tab Menus

How to remove the secondary tabs under the main tabs.

1. In the vtiger home directory, open the parent\_tabdata.php (not used any more since 540)

```
<?php
//This file contains the commonly used variables
$parent_tab_info_array = array(1=>'My Home Page', 2=>'Marketing', 3=>'Sales', 4=>'Support',
5=>'Analytics', 6=>'Inventory', 7=>'Tools', 8=>'Settings');
$parent_child_tab_rel_array = array(1=>array(3,9,10,), 2=>array(26,6,4,10,7,9,8,),
3=>array(7,6,4,2,20,22,23,14,19,8,9,), 4=>array(13,15,6,4,14,8,10,9,), 5=>array(1,25,),
6=>array(14,18,19,21,22,20,23,), 7=>array(24,27,8,), 8=>array(),);
?>
```

\$parent\_tab\_info\_array: You can change the tab order just changing the array order.

If you want some tab to disappear, delete or comment that tab.

2. For example, to "disappear" the Marketing tab and move the Support tab: (not used any more since 540)

```
$parent_tab_info_array = array(1=>'My Home Page', /*2=>'Marketing',*/ 3=>'Sales', 5=>'Analytics',
6=>'Inventory', 4=>'Support', 7=>'Tools', 8=>'Settings');
```

\$parent\_child\_tab\_rel\_array: Each array is related to \$parent\_tab\_info\_array for changing the child\_tab order

you have to change the elements of the array that you want to modify.

3. For My Home Page, (not used any more since 540)

```
$parent_child_tab_rel_array=array( 1=>array(3,9,10,21,),
```

The numbers in the arrays correspond to child tabs, as set in the \$tab\_info\_array array, which is defined in the tabdata.php file.

**4. Important** To finish removing you must delete an entry in the data base. Find the parenttabid in the table vtiger\_parenttab rememeber it, you must find the number of the secondary tabs in tabdata.php \$tab\_info\_array too. Go in table vtiger\_parenttabrel find all record with the parenttabid find before, remove the entry where tabid correspond to your secondary number. Reload your page, the submenu has disappear.

```
delete from vtiger_parenttabrel where parenttabid=2 and tabid=26 and sequence=1; // remove tab
INSERT INTO vtiger_parenttabrel (parenttabid, tabid, sequence) VALUE (2,26,1); // add tab
```

5. How to check the tabid and sequence:

```
mysql> select * from vtiger_tab \g;
```

### Change Home page Components values

```
mysql> select * from vtiger_homestuff where userid="7";
```

stuffid	stuffsequence	stufftype	userid	visible	stufftitle
46	1	Default	7	1	Top Accounts
47	2	Default	7	1	Home Page Dashboard
48	3	Default	7	1	Top Potentials
49	4	Default	7	1	Top Quotes
50	5	Default	7	1	Key Metrics
51	6	Default	7	1	Top Trouble Tickets
52	7	Default	7	1	Upcoming Activities
53	8	Default	7	0	My Group Allocation
54	9	Default	7	1	Top Sales Orders
55	10	Default	7	1	Top Invoices
56	11	Default	7	1	My New Leads
57	12	Default	7	1	Top Purchase Orders
58	13	Default	7	1	Pending Activities
59	14	Default	7	1	My Recent FAQs
60	15	Tag Cloud	7	0	Tag Cloud

15 rows in set (0.00 sec)

```
mysql> update vtiger_homestuff set visible="1" where stufftitle="Top Accounts";
```

- Change default value

## Change homeModComptVisibility variable under modules/Users/Users.php

```
function getDefaultHomeModuleVisibility($home_string,$inVal)
{
    $homeModComptVisibility=0;
    if($inVal == 'postinstall')
    {
        if($_REQUEST[$home_string] != )
        {
            $homeModComptVisibility=1;
        }
    }
    return $homeModComptVisibility;
}
```

### Changing an existing field: type pick-list

Brief explanation of the important variables stored

- **tabid**: ID of the module on which the field is present (from vtiger\_tab)
- **fieldid**: ID of field; generated by getUniqueID("vtiger\_field").
- **columnname**: Name of the column in its table any
- **tablename**: Name of the table that stores field (contactscf-> means contacts custom field)
- **generatedtype**: Specifies type of field in module whether 1='Existing' or 2='User Defined'
- **uitype**: Type of display field (type 33 combo box)
- **fieldname**: The field name generated by Vtiger (it also creates a table with that field name)
- **fieldlabel**: The label of the field important: vTiger will look for fieldlabel as a key in the the \$mod\_strings array stored at modules/ModuleName/language/<language prefix>.lang.php
- **info\_type**: 'BAS' field is displayed in Basic Information, 'ADV' field is displayed in More information BAS or ADV

1. The first step is to find the field you want to convert:

All the fields are stored in the table vtiger\_field. A good parameter to look for is the field label which is the label shown in web page display.

```
SELECT * FROM `vtiger_field` WHERE `tablename` = "dummy combo box"
```

```
tabid: 4
fieldid: 617
columnname: cf_617
```

```

        tablename: vtiger_contactscf
generatedtype: 2
        uitype: 33
        fieldname: cf_617
        fieldlabel: dummy combo box
        readonly: 0
        presence: 2
        defaultvalue:
maximumlength: 100
        sequence: 29
        block: 4
        displaytype: 1
        typeofdata: V~0
        quickcreate: 1
quickcreatesequence: 0
        info_type: BAS
        masseditable: 1
        helpinfo: NULL

```

2. Determine the unique picklist number for this new picklist.

- Table vtiger\_picklist contains the picklist number and also the name (fieldname in vtiger\_field)

```
picklistid: 49
name: cf_617
```

- Table vtiger\_picklist\_seq contains the last known picklist number. If you want to add a new value you need to change this value (incremental)

### 3. Tables on which the picklist values are stored

- Actual values (invoked for each account):

```
mysql> select * from vtiger_accountscf;
```

accountid	cf_615
20	NULL
125	ANA  ##  COMBO  ##  LIFEWATCH
126	ANA
127	ANA  ##  COMBO  ##  LIFEWATCH
128	ANA  ##  COMBO
129	COMBO  ##  LIFEWATCH

130	ANA ## COMBO ## LIFEWATCH
131	LIFEWATCH
132	LIFEWATCH

- Default values:

```
mysql> select * from vtiger_cf_615;
+-----+-----+-----+-----+
| cf_615id | cf_615 | presence | picklist_valueid |
+-----+-----+-----+-----+
| 1 | ANA | 1 | 290 |
| 2 | LIFEWATCH | 1 | 291 |
| 3 | COMBO | 1 | 292 |
+-----+-----+-----+-----+
```

- Add more Default values

```
mysql> select * from vtiger_cf_617_seq;
+-----+
| id |
+-----+
| 5 |
+-----+
```

- Picklist values

It is possible to change values (change names) using the picklist editor tool. Those values are stored in:

```
mysql> select * from vtiger_picklist_dependency \G;
***** 1. row *****
      id: 1
     tabid: 6
sourcefield: industry
targetfield: accounttype
sourcevalue: Banking
targetvalues: ["--None--", "Analyst", "Integrator", "Investor", "Partner", "Press", "Prospect", "Reseller", "Other"]
      criteria: NULL
```

- Examples:

Add a new DEFAULT values into a combo box:

Sequence -> Check in table vtiger\_field the fieldname-> go tho vtiger\_fieldname and create the . variables -> remember to add



the sequence (**important:** the picklist\_valueid is unique for all picklist values)

## How to remove the Java script edit per field

To disable the edit per field it is necessary to change the function hndMouseOver() in include/js/dtlviewajax.js like this:

```
function hndMouseOver(uitype,fieldLabel)
{
return true;
var mouseArea="";
mouseArea="mouseArea_"+ fieldLabel;
...
```

The "return true" at the start of the function turns off all "edit" labels from appearing and thus you cannot get at the ajax section. If you want to customized it you will have to read the module variable and put an "if" condition around the "return true".

## Authorization Policies

### Roles, Profiles and Users

- We have created two different **Roles**: *EGI.eu* and *NIL*.
- Each user must be associated with a unique **Role**.
- Each **Role** inherits the authorization policies defined for one or more **Profiles**. Currently there is a one to one mapping
  1. NIL Role --> NIL Profile
  2. EGI.eu Role --> EGI Member Profile
- The following example shows the NIL Role definition.

```
Role Name:      NIL
Reports to:     Organisation
Members:
  Associated Profiles:
    NIL Profile
  Associated Users:
    Borges
    Ivana Krenkova
    Genevieve Romier
    (...)
```

### Roles, Groups and Users

- There is a **Group** per NGI
- Individual users belonging to the NGI must be included in the **Group**
- **Roles** *EGI.eu* and *NIL* are also included in the **Group** so that other users may modified objects assigned to the group, although they are not implicit **Group** members.
- The following example shows the membership of group *NGI Portugal (PT)*

```
Group Name:    NGI Portugal (PT)
Description:   Group responsible by the Portuguese projects
Members:
  Roles:
    EGI.eu
    NIL
  Users:
    Borges
    Joao Pina
```

## User based ACLs

These instructions allow to access the functionality to set User ACLs

1. Login as Administrator
2. Select *CRM Settings* (top-right menu)
3. Select *Users*
4. Select *Sharing Access*
5. Select *Public: Read, Create/Edit, Delete* in all modules
6. Save
7. Recalculate

## Administrative tasks (user:admin)

### Home page Components

Configurable for each individual user (don't depend on roles / profiles). Admin only selects the first view after each individual users is allowed to change is own Home page by going into ( My Preferences )

- Steps

```
Settings -> Users -> Select user -> edit -> 8. Home page components (show / hide)
```

## Disable Email

Comment the following lines in ("modules/Users/Save.php")

```
// Do not send mails - Goncalo Borges
// $mail_status =
send_mail('Users',$user_emailid,$HELPDESK_SUPPORT_NAME,$HELPDESK_SUPPORT_EMAIL_ID,$subject,$email_body);
// if($mail_status != 1) {
//     $mail_status_str = $user_emailid."=".$mail_status."&&&";
//     $error_str = getMailErrorString($mail_status_str);
// }
```

# Development

## Implemented Debug Output to Screen Functions

Since our knowledge of built-in debug it's short and the debug affects a little the application performance, we made the decision to implement "Debug Functions" to make the debug/deployment process easier.

- Functions ( varshow, debuglog and debugvar )
- function varshow()

```
/** Function used to display variables values highlighted - Output to screen **/
* @param $value -- value:: Type string
* @param $cor -- value:: Type string - w/ color format
* @param $fundo -- value:: Type string - w/ color format

function varshow($value, $cor, $fundo) {
    $fundo =
array('00008B','008B8B','8B008B','8B0000','CD661D','8B4513','CD2626','00868B','104E8B','36648B','00688B',
'4A708B','607B8B','2E8B57');
    if ($cor){
        $cor = $cor;
        $fundo = array_rand($fundo);
    }else{
        $cor = 'white';
        $fundo = array_rand($fundo);
    }
    $valor = "<span style=background:#".$fundo.";color:".$cor.">".$value."</span>";
    echo $valor;
    return;
}
```

**USAGE: varshow(\$variable\_name);**

- function debuglog()

```
/** Function used to log simple variables and blocks of data(strings)
 * @param $data -- value:: Type string, integer
 */
function debuglog($data) {
    $myFile = "vtiger_debug.log";
    $fh = fopen($myFile, 'a') or die("can't open file");
    $stringData = $data."\n";
    fwrite($fh, $stringData);
    fclose($fh);
}
```

**USAGE: debuglog(\$variablename.'String used to identify') or just debuglog(\$variablename);**

- function debugvar()

```
/** Function used to log any type of variables
 * @param $var -- value:: Type string, integer
 * @param $label -- value:: Type string, integer
 * @param $exit -- value:: Type boolean
 */
function debugvar($var, $label, $exit = false) {

    if (is_array($var) || is_object($var)) {
        $data = "ARRAY";
        $data .= htmlentities(print_r($var, true));
    }
    elseif (is_string($var)) {
        $data = "STRING";
        $data .= "string(" . strlen($var) . ") \"" . htmlentities($var) . "\"\n";
    }
    else {
        $data = "VAR_DUMP";
        $data .= var_dump($var);
    }
    $myFile = "vtiger_debug.log";
    $fh = fopen($myFile, 'a') or die("can't open file");
    $stringData = $label.":\n".$data."\n";
```

```
        fwrite($fh, $stringData);  
        fclose($fh);  
        if ($exit) {  
            exit;  
        }  
    }
```

**USAGE:** `debugvar($variablename,'Label with text to help identify variable on the log (Vartype)');`

# Operation

## MySQL Backups

Backup is implemented through cronjobs running three times a day (at 3,13,18) a full database ("--all databases") incremental backup.

The *Production server* "vt.my.domain" make is own backup and the *Development server* "vtdev.my.domain" make is own backup, plus the copies of production dumps for some redundancy.

### VT.MY.DOMAIN

Backup Location: /var/www/backups (filename ex: vtiger-producao-mysql\_DB\_D\_2012-03-26-H\_03.bk.gz)

vtiger-mysql\_DB.bk

---

### VTDEV.MY.DOMAIN

Backup Location: Development - /var/www/backups (filename: vtiger-mysql\_DB\_D\_2012-03-22-H\_03.bk.gz) Production - /var/www/backups/producao (filename: vtiger-producao-mysql\_DB\_D\_2012-03-26-H\_03.bk.gz)

## Servers Management

User specific aliases and functions

### [Common]

alias **backups**='cd /var/www/backups'

alias **repos**='cd /var/www/repository'

**[Development] - vtdev.my.domain**

alias **devcm**='cd /var/www/html/vtbranch-5-3-0-devcm'

alias **devjp**='cd /var/www/html/vtbranch-5-3-0-devjp'

alias **dev**='cd /var/www/html/vtbranch-5-3-0-dev'

**[Production] - vt.my.domain**

alias **production**='cd /var/www/html/vtigercrm'

alias **dev**='cd /var/www/html/vtdev' (Pre-production instance)

## HTTPD Configuration (in production)

```
# cat /etc/httpd/conf/httpd.conf

<VirtualHost vt.my.domain:80>
ServerAdmin user@my.domain
ServerName vt.my.domain
DocumentRoot /var/www/html/vtigercrm/
ErrorLog logs/vt.my.domain-error_log
CustomLog logs/vt.my.domain-access_log common
ServerAlias
#Redirect / http://vt.my.domain/vtigercrm
#Redirect          /vtigercrm          "https://vt.my.domain"
Redirect          /                    "https://crm.egi.eu"
</VirtualHost>

Alias /vtdev/ "/var/www/html/vtdev/"
Alias /vtdev "/var/www/html/vtdev/"

Alias /vtigercrm/ "/var/www/html/vtigercrm/"
Alias /vtigercrm "/var/www/html/vtigercrm/"

Alias /vtdev-db/ "/var/www/html/vtdev-cluster/"
Alias /vtdev-db "/var/www/html/vtdev-cluster/"
```

## Utilities

### mysqlvtigercrm

script para efectuar backups e listagem das bases de dados. argumentos: --all, --db, --list --all : faz o backup de todas as bases de dados existentes no servidor para o directorio actual. --db : faz o backup da bases de dados especificada para o directorio actual. --

list : mostra uma lista das bases de dados existentes no servidor

USAGE: mysqlvtcrm %command% args

```
mysqlvtcrm --all
mysqlvtcrm --db dbname
mysqlvtcrm --list
```

## vtigerepos ""(Not finished)""

Script to manage vtiger local repositories. PATH: /var/www/repository

Possible operations/commands - backup, debug, rollback, list

USAGE: vtigerepos %command% args

```
vtigerepos backup %user% - users: pina, cnm or other defined
vtigerepos debug %pre|post% - 1|2
vtigerepos rollback or rollback %filename% - attention: rollback without args will rollback to the last changed file
vtigerepos list or list %ol% - only last file
```

## How to enable Table Relation View in PHPmyadmin

1. Create a phpuser and give it the correct permissions in the mysql.

```
GRANT USAGE ON mysql.* TO 'PHPAdmin'@'localhost' IDENTIFIED BY 'password';
GRANT SELECT (
    Host, User, Select_priv, Insert_priv, Update_priv, Delete_priv,
    Create_priv, Drop_priv, Reload_priv, Shutdown_priv, Process_priv,
    File_priv, Grant_priv, References_priv, Index_priv, Alter_priv,
    Show_db_priv, Super_priv, Create_tmp_table_priv, Lock_tables_priv,
    Execute_priv, Repl_slave_priv, Repl_client_priv
) ON mysql.user TO 'PHPAdmin'@'localhost';
GRANT SELECT ON mysql.db TO 'PHPAdmin'@'localhost';
GRANT SELECT ON mysql.host TO 'PHPAdmin'@'localhost';
GRANT SELECT (Host, Db, User, Table_name, Table_priv, Column_priv)
ON mysql.tables_priv TO 'PHPAdmin'@'localhost';
```

2. edit config.inc.php in the phpmyadmin machine

```
$cfg['Servers'][$i]['pmadb'] = 'phpmyadmin';
```

```
$cfg['Servers'][$i]['bookmarktable'] = 'pma_bookmark';
$cfg['Servers'][$i]['relation'] = 'pma_relation';
$cfg['Servers'][$i]['table_info'] = 'pma_table_info';
$cfg['Servers'][$i]['table_coords'] = 'pma_table_coords';
$cfg['Servers'][$i]['pdf_pages'] = 'pma_pdf_pages';
$cfg['Servers'][$i]['column_info'] = 'pma_column_info';
$cfg['Servers'][$i]['history'] = 'pma_history';
```

### 3. Create the following tables and also SET SQL\_MODE:

```
- phpMyAdmin SQL Dump
-- version 2.11.10
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Jan 04, 2012 at 11:29 AM
-- Server version: 5.5.17
-- PHP Version: 5.3.8
```

```
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
```

```
--
-- Database: `phpmyadmin`
--
```

```
-- -----
```

```
--
-- Table structure for table `pma_bookmark`
--
```

```
CREATE TABLE IF NOT EXISTS `pma_bookmark` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `dbase` varchar(255) COLLATE utf8_bin NOT NULL DEFAULT ,
  `user` varchar(255) COLLATE utf8_bin NOT NULL DEFAULT ,
  `label` varchar(255) CHARACTER SET utf8 NOT NULL DEFAULT ,
  `query` text COLLATE utf8_bin NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='Bookmarks' AUTO_INCREMENT=1 ;
```

```
-- -----
```



```
--
-- Table structure for table `pma_column_info`
--

CREATE TABLE IF NOT EXISTS `pma_column_info` (
  `id` int(5) unsigned NOT NULL AUTO_INCREMENT,
  `db_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `table_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `column_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `comment` varchar(255) CHARACTER SET utf8 NOT NULL DEFAULT ,
  `mimetype` varchar(255) CHARACTER SET utf8 NOT NULL DEFAULT ,
  `transformation` varchar(255) COLLATE utf8_bin NOT NULL DEFAULT ,
  `transformation_options` varchar(255) COLLATE utf8_bin NOT NULL DEFAULT ,
  PRIMARY KEY (`id`),
  UNIQUE KEY `db_name` (`db_name`,`table_name`,`column_name`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='Column information for phpMyAdmin'
AUTO_INCREMENT=1 ;

-----

--
-- Table structure for table `pma_designer_coords`
--

CREATE TABLE IF NOT EXISTS `pma_designer_coords` (
  `db_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `table_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `x` int(11) DEFAULT NULL,
  `y` int(11) DEFAULT NULL,
  `v` tinyint(4) DEFAULT NULL,
  `h` tinyint(4) DEFAULT NULL,
  PRIMARY KEY (`db_name`,`table_name`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='Table coordinates for Designer';

-----

--
-- Table structure for table `pma_history`
--

CREATE TABLE IF NOT EXISTS `pma_history` (
  `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
  `username` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
```

```
`db` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
`table` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
`timevalue` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
`sqlquery` text COLLATE utf8_bin NOT NULL,
PRIMARY KEY (`id`),
KEY `username` (`username`,`db`,`table`,`timevalue`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='SQL history for phpMyAdmin' AUTO_INCREMENT=52 ;

--
--
-- Table structure for table `pma_pdf_pages`
--

CREATE TABLE IF NOT EXISTS `pma_pdf_pages` (
  `db_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `page_nr` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `page_descr` varchar(50) CHARACTER SET utf8 NOT NULL DEFAULT ,
  PRIMARY KEY (`page_nr`),
  KEY `db_name` (`db_name`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='PDF relation pages for phpMyAdmin'
AUTO_INCREMENT=1 ;

--
--
-- Table structure for table `pma_relation`
--

CREATE TABLE IF NOT EXISTS `pma_relation` (
  `master_db` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `master_table` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `master_field` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `foreign_db` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `foreign_table` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `foreign_field` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  PRIMARY KEY (`master_db`,`master_table`,`master_field`),
  KEY `foreign_field` (`foreign_db`,`foreign_table`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='Relation table';

--
--
```

```
-- Table structure for table `pma_table_coords`
--

CREATE TABLE IF NOT EXISTS `pma_table_coords` (
  `db_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `table_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `pdf_page_number` int(11) NOT NULL DEFAULT '0',
  `x` float unsigned NOT NULL DEFAULT '0',
  `y` float unsigned NOT NULL DEFAULT '0',
  PRIMARY KEY (`db_name`,`table_name`,`pdf_page_number`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='Table coordinates for phpMyAdmin PDF output';

-----

--
-- Table structure for table `pma_table_info`
--

CREATE TABLE IF NOT EXISTS `pma_table_info` (
  `db_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `table_name` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  `display_field` varchar(64) COLLATE utf8_bin NOT NULL DEFAULT ,
  PRIMARY KEY (`db_name`,`table_name`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin COMMENT='Table information for phpMyAdmin';
```

## CheckList for Upgrading

- Re-implement Google Analytics
- Re-implement LDAP authentication
- Re-implement Dashboards Code
- Re-check Workflows
- Re-deploy Metrics scripts
- Disable sending email
- Check the Web Form Validation
- Check the Web Displays
- Check if the *Search Filters* are working properly
- Check if the *Reports* are working properly

## **Links**

---

---

last edited 2014-05-07 12:40:43 by **goncalo**