Diario di lavoro

|  |  |
| --- | --- |
| Luogo | Arti Mestieri Trevano |
| Data | 08.11.2019 |

|  |
| --- |
| Lavori svolti |
| Ho fatto la convalidazione lato client dell’utente, ho fatto le funzioni di convalidazione lato server:  public function validatePhoneNumber($phoneNumber) {  $validElement = $this->generalValidation($phoneNumber);  $pattern = '/^[a-zèéëàáäìíòöóüùú\s]\*$/i';   if (*strlen*($phoneNumber) > 0 && *strlen*($phoneNumber) <= 45 && *preg\_match*($pattern, $validElement)) {  return true;  } else {  return false;  } } public function validateString($string){  $validElement = $this->generalValidation($string);  $pattern = '/^[0-9\s+#]+$/i';   if (*strlen*($validElement) > 0 && *strlen*($validElement) <= 45 && *preg\_match*($pattern, $validElement)) {  return true;  } else {  return false;  } } public function validatePermission($permission){  if($permission == "User" || $permission == "Admin"){  return true;  }else{  return false;  } } public function validateEmail($email){  $validElement = $this->generalValidation($email);  $pattern = '/^(([^<>()[\]\\.,;:\s@\"]+(\.[^<>()[\]\\.,;:\s@\"]+)\*)|(\".+\"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/';   if( *preg\_match*($pattern, $validElement)){  return true;  }else{  return false;  } }  Poi ho iniziato ad implementare la validazione dell’utente con le varie funzioni e nel mentre mi sono venute un po’ di idee per ottimizzare il codice quindi mi sono messo a fare quello.  Ho ottimizato il metodo per fare il require della pagina in questo modo ho solo una funzione invece di averne due che facevano la stessa cosa:  public function requirePageForm($pageInformation){  $title = $pageInformation[0];  $nameButton = $pageInformation[1];  $path = $pageInformation[2];  $stringErrors = $pageInformation[3];  $name = null;  if(*count*($pageInformation) > 4){  $name = $pageInformation[4];  }  if($this->arrayTank!=null){  $tankName = $this->arrayTank["tankName"];  $calcium = $this->arrayTank["calcium"];  $magnesium = $this->arrayTank["magnesium"];  $kh = $this->arrayTank["kh"];  $waterChange = $this->arrayTank["waterChange"];  $liter = $this->arrayTank["liter"];  }else if($name != null){  $tankToModify = $this->tankManagementModel->getByName($name);  $tankName = $tankToModify[0]["nome"];  $magnesium = $tankToModify[0]["calcio"];  $calcium = $tankToModify[0]["magnesio"];  $kh = $tankToModify[0]["kh"];  $waterChange = $tankToModify[0]["ultimo\_cambio\_acqua"];  $liter = $tankToModify[0]["Litri"];  }   require "GestioneAcquariMarini/views/\_templates/header.php";  require "GestioneAcquariMarini/views/\_templates/menu.php";  require "GestioneAcquariMarini/views/gestioneAcquari/tank/tankManagement.php";  require "GestioneAcquariMarini/views/\_templates/footer.php"; } |

|  |
| --- |
| Problemi riscontrati e soluzioni adottate |
| Array ( [tankName] => zzzz [magnesium] => 0 [calcium] => 0 [kh] => 20 [waterChange] => 2000-11-11 [liter] => 33 ) vasca1 Fatal error: Uncaught PDOException: SQLSTATE[23000]: Integrity constraint violation: 1062 Duplicate entry '' for key 'PRIMARY'  Mi sono accorto che quando modifico la vasca e modifico anche il nome cioè la primary key mi dice che ce un errore e che la primary key è duplicata, solo che non capisco qual’è il problema dato che il codice che ho scritto è giusto dato che facendo la stessa cosa trami interfacci grafica da MySQL Workbanch il codice utilizzato è lo stesso quindi non capisco qual’è il problema. |

|  |
| --- |
| Punto della situazione rispetto alla pianificazione |
| In anticipo di una settimana |

|  |
| --- |
| Programma di massima per la prossima giornata di lavoro |
| -Finire la convalidazione lato server per gli utenti |