Diario di lavoro

|  |  |
| --- | --- |
| Luogo | Arti Mestieri Trevano |
| Data | 07.11.2019 |

|  |
| --- |
| Lavori svolti |
| Oggi ho finito la convalidazione lato server e client della aggiunta e modifica delle vasche. Infatti ho modificato due cosette come l’aggiunta di un minimo e un massimo di caratteri per le stringhe e per i numeri, li ho ritestati poi ho fatto il backend.  Ho creato 2 classi per farlo:  ValidationFunction => in cui metto le funzioni generali  TankValidation => in cui faccio la validazione della vasca che sta per essere aggiunta o modificata  TankValidation:  class TankValidation {  private $validationFunction;  public $stringErrors;   public function \_\_construct()  {  require\_once "validationFunction.php";  $this->validationFunction = new ValidationFunction();  }   public function validation($tank){  $validationOK = true;  $this->stringErrors = "";   if(!$this->validationFunction->validateTankName($tank["tankName"])){  $this->stringErrors = "Il nome dell'acquario è sbagliato";  $validationOK = false;  }  if(!$this->validationFunction->validateInt($tank["magnesium"], 0, 3000)){  $this->stringErrors .= "<br>Il valore del magnesio è sbagliato";  $validationOK = false;  }  if(!$this->validationFunction->validateInt($tank["calcium"], 0, 1000)){  $this->stringErrors .= "<br>Il valore del calcio è sbagliato";  $validationOK = false;  }  if(!$this->validationFunction->validateInt($tank["kh"], 0, 20)){  $this->stringErrors .= "<br>Il valore del kh è sbagliato";  $validationOK = false;  }  if(!$this->validationFunction->validateDate($tank["waterChange"])){  $this->stringErrors .= "<br>La data è sbagliata";  $validationOK = false;  }  if(!$this->validationFunction->validateInt($tank["liter"], 0, 1000000)){  $this->stringErrors .= "<br>Il litraggio della vasca è sbagliato";  $validationOK = false;  }   return $validationOK;  } }  Ho modificato anche il controller tankManagement:  Questo metodo fa il require del form per aggiungere le vasche.  public function requirePageAddForm(){  $title = "Aggiungi vasca";  $nameButton = "Aggiungi";  $path = *URL*."tankManagement/addTank";  $stringErrors = $this->tankValidatioModel->stringErrors;   if($this->arrayTank!=null){  $tankName = $this->arrayTank["tankName"];  $calcium = $this->arrayTank["calcium"];  $magnesium = $this->arrayTank["magnesium"];  $kh = $this->arrayTank["kh"];  $waterChange = $this->arrayTank["waterChange"];  $liter = $this->arrayTank["liter"];  }   require "GestioneAcquariMarini/views/\_templates/header.php";  require "GestioneAcquariMarini/views/\_templates/menu.php";  require "GestioneAcquariMarini/views/gestioneAcquari/tank/tankManagement.php";  require "GestioneAcquariMarini/views/\_templates/footer.php"; }  Questo è il metodo richiamato quando l’utente preme il bottone aggiungi  public function formAddTank(){  *session\_start*();  if($\_SESSION["authentification"] == true){  $this->requirePageAddForm();  }else{  *header*("Location:".*URL*);  } }  Metodo che mi ritorna i valori negli input:  private function getTankArray(){  $name = $\_POST["tankName"];  $magnesium = $\_POST["magnesium"];  $calcium = $\_POST["calcium"];  $kh = $\_POST["kh"];  $waterChange = $\_POST["waterChange"];  $liter = $\_POST["liter"];   $tank = array("tankName"=>$name, "magnesium"=>$magnesium, "calcium"=>$calcium, "kh"=>$kh, "waterChange"=>$waterChange, "liter"=>$liter);  return $tank; }  Nel tankModel ho aggiunto una funzione che mi ritorna tutti i nomi delle vasche presenti sul database per evitare che venga inserita un nome già esistente:  public function getAllTankName(){  $selectThank = "select nome from vasca";  $allName = $this->executeAndFetchStatement($selectThank);  return $allName; }  Nella classe validation ho aggiunto e modificato in particolare queste funzioni:  public function validateTankName($tankName) {  require\_once "tankModel";  $tankModel = new TankModel();  $validElement = $this->generalValidation($tankName);  $pattern = '/^[A-Za-z0-9\_-]\*$/';  if (*preg\_match*($pattern, $validElement) && *strlen*($validElement) > 0 && *strlen*($validElement) <= 45) {  $allName = $tankModel->getAllTankName();  if (!*in\_array*($tankName, $allName)){  return true;  }else{  return false;  }  }else{  return false;  } }  public function validateDate($date){  $date\_arr = *explode*('-', $date);  $current\_date = new DateTime();  $dateOk = false;  if (*count*($date\_arr) == 3) {  if (*checkdate*($date\_arr[2], $date\_arr[1], $date\_arr[0])) {  if ($date > $current\_date) {  $dateOk = true;  }  }  }  return $dateOk; } |

|  |
| --- |
| Problemi riscontrati e soluzioni adottate |
| - |

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

|  |
| --- |
| Programma di massima per la prossima giornata di lavoro |
| -Fare la convalidazione lato client e server per gli utenti |