*Arhitektuuri dokument*

Süsteemi kirjeldus



**Dokumendi ajalugu**

|  |  |  |  |
| --- | --- | --- | --- |
| **Kuupäev** | **Versioon** | **Muudatuse kirjeldus** | **Autor** |
| 12.06.2019 | 0.1 | Esialgne versioon | Rasmus Aron |
| 18.06.2019 | 0.2 | Teksti korrigeerimine | Rasmus Aron |
| 20.06.2019 | 1.0 | Viimistlemine | Rasmus Aron |
|  |  |  |  |
|  |  |  |  |

Sisukord

[1 Sissejuhatus 7](#_Toc11922941)

[1.1 Akronüümid/lühendid, mõisted 7](#_Toc11922942)

[1.2 Taust 7](#_Toc11922943)

[1.3 Dokumendi Skoop 7](#_Toc11922944)

[1.4 Components 8](#_Toc11922945)

[1.4.1 ActiveForm 8](#_Toc11922946)

[1.4.1.1 function begin($method = "POST", $enctype = null, $action = null) 8](#_Toc11922947)

[1.4.1.2 function end() 8](#_Toc11922948)

[1.4.1.3 function submitButton($btnText, $options) 8](#_Toc11922949)

[1.4.1.4 function \_\_toString() 8](#_Toc11922950)

[1.4.1.5 function label($labelVal) 8](#_Toc11922951)

[1.4.1.6 function field($model, $fieldName) 8](#_Toc11922952)

[1.4.1.7 function password() 8](#_Toc11922953)

[1.4.1.8 function dropDownList 8](#_Toc11922954)

[1.4.1.9 function checkBox() 8](#_Toc11922955)

[1.4.1.10 function textarea() 8](#_Toc11922956)

[1.4.2 ActiveRecord 9](#_Toc11922957)

[1.4.2.1 function find() 9](#_Toc11922958)

[1.4.2.2 function addWhere($operator, $fieldName, $fieldValue) 9](#_Toc11922959)

[1.4.2.3 function orWhere($operator, $fieldName, $fieldValue) 9](#_Toc11922960)

[1.4.2.4 function one() 9](#_Toc11922961)

[1.4.2.5 function all() 9](#_Toc11922962)

[1.4.2.6 function allNames() 9](#_Toc11922963)

[1.4.2.7 function allUniqueNames() 9](#_Toc11922964)

[1.4.3 Alert 9](#_Toc11922965)

[1.4.3.1 function showMessages() 9](#_Toc11922966)

[1.4.4 BaseController 9](#_Toc11922967)

[1.4.4.1 function initialAction($action, $controller, $params) 10](#_Toc11922968)

[1.4.4.2 function handleBehaviors($instance) 10](#_Toc11922969)

[1.4.4.3 function render($viewName, $params = []) 10](#_Toc11922970)

[1.4.4.4 function renderPartial($viewName, $params = []) 10](#_Toc11922971)

[1.4.4.5 function redirect($action, $params = []) 10](#_Toc11922972)

[1.4.4.6 function goHome() 10](#_Toc11922973)

[1.4.4.7 function requireToVar($viewName, $params) 10](#_Toc11922974)

[1.4.4.8 function json($data) 10](#_Toc11922975)

[1.4.5 BaseModel 10](#_Toc11922976)

[1.4.5.1 function \_\_toString() 10](#_Toc11922977)

[1.4.5.2 function attributeLabels() 10](#_Toc11922978)

[1.4.5.3 function \_\_get 11](#_Toc11922979)

[1.4.5.4 function \_\_set 11](#_Toc11922980)

[1.4.5.5 function addError($message) 11](#_Toc11922981)

[1.4.5.6 function showErrorsAsHtml() 11](#_Toc11922982)

[1.4.5.7 function save() 11](#_Toc11922983)

[1.4.5.8 function beforeValidate() 11](#_Toc11922984)

[1.4.5.9 function autoFillFields() 11](#_Toc11922985)

[1.4.5.10 public function beforeSave() 11](#_Toc11922986)

[1.4.5.11 public function afterSave() 11](#_Toc11922987)

[1.4.5.12 public function beforeDelete() 11](#_Toc11922988)

[1.4.5.13 function setValueToAllRuleFields($fields, $value, $byPassEmpty = false) 11](#_Toc11922989)

[1.4.5.14 function validate() 11](#_Toc11922990)

[1.4.5.15 function resetAllAttributes() 11](#_Toc11922991)

[1.4.5.16 function load($post) 12](#_Toc11922992)

[1.4.5.17 function delete() 12](#_Toc11922993)

[1.4.5.18 function hasErrors() 12](#_Toc11922994)

[1.4.5.19 function validateEmail($email) 12](#_Toc11922995)

[1.4.6 Detail View 12](#_Toc11922996)

[1.4.6.1 static function widget($data) 12](#_Toc11922997)

[1.4.7 Flash 12](#_Toc11922998)

[1.4.7.1 function get() 12](#_Toc11922999)

[1.4.7.2 function setMessage($index, $message) 12](#_Toc11923000)

[1.4.7.3 function removeMessage($index) 12](#_Toc11923001)

[1.4.7.4 function getMessage($index) 12](#_Toc11923002)

[1.4.7.5 function hasMessage($index) 12](#_Toc11923003)

[1.4.7.6 function getMessages() 12](#_Toc11923004)

[1.4.7.7 function unsetMessages() 13](#_Toc11923005)

[1.4.8 GridView 13](#_Toc11923006)

[1.4.8.1 function widget($data) 13](#_Toc11923007)

[1.4.9 Helper 13](#_Toc11923008)

[1.4.9.1 function generateRandomString($length = 128) 13](#_Toc11923009)

[1.4.9.2 function getDegrees() 13](#_Toc11923010)

[1.4.9.3 function generateSemesters($howManyYears = 3) 13](#_Toc11923011)

[1.4.9.4 function sendMail($from, $to, $subject, $message) 13](#_Toc11923012)

[1.4.9.5 function setTitle($name) 13](#_Toc11923013)

[1.4.9.6 function getTitle() 13](#_Toc11923014)

[1.4.9.7 function getClassName($obj) 13](#_Toc11923015)

[1.4.9.8 function isStringClean($str, $maxLength = 128) 13](#_Toc11923016)

[1.4.9.9 function getCountries() 13](#_Toc11923017)

[1.4.10 Identity 13](#_Toc11923018)

[1.4.10.1 function isLoggedIn() 14](#_Toc11923019)

[1.4.10.2 function getUserId() 14](#_Toc11923020)

[1.4.10.3 function isGuest() 14](#_Toc11923021)

[1.4.10.4 function validateIdentity() 14](#_Toc11923022)

[1.4.10.5 function login($user) 14](#_Toc11923023)

[1.4.10.6 function logout() 14](#_Toc11923024)

[1.4.10.7 function &get() 14](#_Toc11923025)

[1.4.11 QueryBuilder 14](#_Toc11923026)

[1.4.11.1 function \_\_construct($tableName) 14](#_Toc11923027)

[1.4.11.2 function select($tableName) 14](#_Toc11923028)

[1.4.11.3 function addWhere($operator, $fieldName, $fieldValue) 14](#_Toc11923029)

[1.4.11.4 function orWhere($operator, $fieldName, $fieldValue) 14](#_Toc11923030)

[1.4.11.5 function refValues($arr) 14](#_Toc11923031)

[1.4.11.6 function composeParams($fieldValues) 14](#_Toc11923032)

[1.4.11.7 function compose($limit = 0) 15](#_Toc11923033)

[1.4.11.8 function execute() 15](#_Toc11923034)

[1.4.11.9 function query() 15](#_Toc11923035)

[1.4.11.10 function queryAll() 15](#_Toc11923036)

[1.4.11.11 function insert($tableName, $data) 15](#_Toc11923037)

[1.4.11.12 function update($tableName, $data, $condition) 15](#_Toc11923038)

[1.4.11.13 function delete($tableName, $condition) 15](#_Toc11923039)

[1.4.12 Request 15](#_Toc11923040)

[1.4.12.1 function InitializeIfNotSet() 15](#_Toc11923041)

[1.4.12.2 function setParams($params) 15](#_Toc11923042)

[1.4.12.3 function setCookie($name, $value) 15](#_Toc11923043)

[1.4.12.4 function setHeaders($headers) 15](#_Toc11923044)

[1.4.12.5 function setUserAgent($userAgent) 15](#_Toc11923045)

[1.4.12.6 function setUserIP() 16](#_Toc11923046)

[1.4.12.7 function setController($controller) 16](#_Toc11923047)

[1.4.12.8 function setAction($action) 16](#_Toc11923048)

[1.4.12.9 function getParams() 16](#_Toc11923049)

[1.4.12.10 function getCookies() 16](#_Toc11923050)

[1.4.12.11 function getHeaders() 16](#_Toc11923051)

[1.4.12.12 function getUserAgent() 16](#_Toc11923052)

[1.4.12.13 function getUserIP() 16](#_Toc11923053)

[1.4.12.14 function getController() 16](#_Toc11923054)

[1.4.12.15 function getAction() 16](#_Toc11923055)

[1.4.12.16 function applyHeaders() 16](#_Toc11923056)

[1.4.13 Router 16](#_Toc11923057)

[1.4.13.1 function parseUrl() 16](#_Toc11923058)

[1.5 Config 17](#_Toc11923059)

[1.5.1 App 17](#_Toc11923060)

[1.5.2 Db 17](#_Toc11923061)

[1.6 Controllers 17](#_Toc11923062)

[1.6.1 Action 17](#_Toc11923063)

[1.6.1.1 ActionIndex 17](#_Toc11923064)

[1.6.1.2 actionCreate 17](#_Toc11923065)

[1.6.1.3 actionUpdate 17](#_Toc11923066)

[1.6.1.4 actionView 17](#_Toc11923067)

[1.6.1.5 actionSave 18](#_Toc11923068)

[1.6.1.6 actionRemove 18](#_Toc11923069)

[1.6.1.7 actionGet\*Subelement\*(sõltub tabelist) 18](#_Toc11923070)

[1.6.1.8 findModel 18](#_Toc11923071)

[1.6.2 UserController 18](#_Toc11923072)

[1.7 Models 18](#_Toc11923073)

[1.8 Web 18](#_Toc11923074)

[1.8.1 .htaccess 18](#_Toc11923075)

[1.8.2 Autoloader 18](#_Toc11923076)

[1.8.3 Index.php 18](#_Toc11923077)

[1.8.4 Main.js 18](#_Toc11923078)

# 

# Sissejuhatus

Projekti eesmärk on arendada välja Tallinna Ülikoolile standardne partnerülikoolidega tutvumiseks ja valimiseks loodud veebirakendus kasutades selleks PHP-t ja MariaDB ning liidestada rakendus Tallinna Ülikooli veebilehega.

Rakenduse vastutav ja volitatud kasutaja on Tallinna Ülikool.

Arendus on teostatud Tallinna Ülikooli Tarkvaraarenduse praktika (IFI6213.DT) 2019. A kevadsemestri raames.

## Akronüümid/lühendid, mõisted

|  |  |
| --- | --- |
| ***Akronüüm/lühend*** | ***Selgitus*** |
|  |  |

## Taust

## Dokumendi Skoop

Dokument on loodud eesmärgiga seletamaks rakenduse arhitektuurilist ülesehitust, edasiste arenduste ja täienduste lihtsustamiseks.

## Components

### ActiveForm

ActiveFormi funktsioone kasutatakse formide ja inputide käsitlemiseks ja välja kuvamiseks.

#### function begin($method = "POST", $enctype = null, $action = null)

Loob activeform objekti funktsiooni skoobis, kuvab välja html form html elemendi ja tagastab loodud objekti.

#### function end()

Kuvab välja html form lõputagi.

#### function submitButton($btnText, $options)

Loob submit nupu. Funktsioonis olev optionName on HTMLi atribuut ja optionValue on HTMLi väärtus,väärtus sõltub atribuutist ja võib olla vabalt valitud formaadis nt. String, int vms

#### function \_\_toString()

Arvestab kõiki objektidele antud omadusi ja teeb sellest teeb HTMLi tagide kogumiku ning tagastab selle stringi kujul.

#### function label($labelVal)

Määrab ära labeli true/false puhul ja tagastab iseenda uuendatud kujul.

#### function field($model, $fieldName)

Tuleb sisse model – lähtmeandemete asukoht millest hakatakse andmeid töötlema.

Määratakse elementType ja inputType default väärtused.

#### function password()

Määrab ära parooli omaduse ja tagastab iseenda uuendatud kujul.

#### function dropDownList

Options sisse läheb list atribuutidest, mis läheb select tag-i sisse.

optionValAsDataVal määrab ära toStringis innerText HTML option tagi väärtuse.

#### function checkBox()

Määrab ära checkboxi omaduse ja tagastab iseenda uuendatud kujul.

#### function textarea()

Määrab ära textarea omaduse ja tagastab iseenda uuendatud kujul.

### ActiveRecord

Kõik funktsioonid ActiveRecord-is kasutavad QueryBuilderi funktsioonalsust.

ActiveRecords lihtsutab kõikide mudelitel andmebaasid andmete kätte saamist.

#### function find()

Teeb activeRecord objekti ja lisab tehtud objekti sisse querybuilderi objekti on kaasatud mudeli tabeli nimi. Võtab klassinime, kust seest findi kutsuti ja paneb selle activeRecordi objekti. Funktsioon tagastab activerecordi objekti.

#### function addWhere($operator, $fieldName, $fieldValue)

Abifunktsiooni mis kasutab QueryBuilderit. Uuendatud QueryBuilderi objekt määratakse ActiveRecordi queryBuilderi uueks objektiks.

#### function orWhere($operator, $fieldName, $fieldValue)

Abifunktsiooni mis kasutab QueryBuilderit. Uuendatud QueryBuilderi objekt määratakse ActiveRecordi queryBuilderi uueks objektiks.

#### function one()

Käivitab queryBuilderi päringu ja loob mudelist objekti, millest funktsioon find kutsuti ja querybuilderist saadud tulemus pannakse mudeli objekti sisse kasutades load funktsiooni. Tagastab mudeli objekti.

#### function all()

Luuakse array kuhu sisse pannakse nii mitu mudeli objekti , kui quesryBuilderi queryall funktsioon tagastab ja mudelite array tagastatakse all funktsiooni.

#### function allNames()

Luuakse array kuhu sisse pannakse nii mitu mudeli objekti , kui quesryBuilderi queryall funktsioon tagastab ja array indeks on mudelite ID ning toString funktsiooni väärtuse.

#### function allUniqueNames()

Luuakse array kuhu sisse pannakse nii mitu mudeli objekti , kui quesryBuilderi queryall funktsioon tagastab ja array indeks on mudelite ID ning toString funktsiooni väärtuse. Lõpetuseks eemaldadakse kanded mille toString värrtused olid samad.

### Alert

#### function showMessages()

Tagastab kõik teavitused HTMLi kujul ja nullib flash sõnumite nimekirja.

### BaseController

Selle projekti selgroog, kõik päringud käivad sisse-välja läbi BaseControlleri.

#### function initialAction($action, $controller, $params)

Seatakse sissetuleva kasutaja metaandmed ja salvestatakse ajutiselt URL-i täpsemad parameetrid. Kasutaja autentitakse. Selgitatakse välja milline Controlleri tegevus peaks järgmise käivituma. Luuakse käivitatavas Controlleris objekt. Kontrollitakse kasutaja õigusi Controlleri ja tegevusi juurdepääsemiseks. Ligipääsu olemasolu käivitatakse vastava Controlleri tegevus.

#### function handleBehaviors($instance)

Kontrollitakse erinevaid behaviored nt. Kas kasutaja on sisseloginud.

#### function render($viewName, $params = [])

Võtab baseView ja subView ning liidab kokku, seejärel saadab sisse tulnud parameetrid alamvaadetesse.

#### function renderPartial($viewName, $params = [])

Sisaldab ainult sub-view parameetreid ja saadab need alamvaatesse. Baseview ei panda alamvaatesse.

#### function redirect($action, $params = [])

PHP teeb https ümbersuunamist teisele lehele.

#### function goHome()

Suunab kasutaja rakenduse kodulehele.

#### function requireToVar($viewName, $params)

päritakse alamvaate fail ning lisatakse parameetrid kaasa. Kuvatakse välja alamvaate tulemus.Kui faili ei leitud tagastakse veateade.

#### function json($data)

Kuvab välja sissetulnud andmestiku.

### BaseModel

#### function \_\_toString()

see funktsioon käivitatakse kui Modeli objektist üritatakse teha stringi, ehk siis näiteks kui echotakse objekti või siis tehakse $obj." mis iganes muu string", siis käivitatakse $obj peal \_\_toString meetod ja tullakse siis ja stringiks on siis see mis $str väärtus on.

#### function attributeLabels()

iin hoitakse andmevälja inimloetavaid kujusid. Model mis extendib BaseModelit omab reaalseid andmevälju ja nendele vastavaid stringe. GridView ja DetailView classid kasutavad neid. Mil antakse andmevälja nimetus ette ja võetakse sellega seotuna siit funktsiooni kaudu vastavad mida välja kuvatakse. See funktsioon tühjalt on lihtsalt siin, et välja kutsumisel errorit ei tuleks kui mõnel klassil ei peaks olema attributeLabels funktsiooni Model klassi sees.

#### function \_\_get

Võimadab pärida andmevälju sellisel kujul $model->mis\_iganes\_andmevali

#### function \_\_set

Võimadab settida andmevälju sellisel kujul $model->mis\_iganes\_andmevali = 5

#### function addError($message)

Veateate lisamise funktsioon.

#### function showErrorsAsHtml()

Võtab kõik enda modeli errorid kokku ja lööb need HTML kujule, et erinevates kohtades seda välja printida mugavalt.

#### function save()

Salvestame andmebaasi SQL kujul päringu

#### function beforeValidate()

getime Model klassis olevad rulesid ja paneme need BaseModeli $this->rules sisse. Kuna autofillFields vajab rulesid.

Paneme sisselogitud kasutaja ID andmevälja väärtuseks.

#### function autoFillFields()

Võimldab kasutada eelnevalt sisetatud informatsiooni.

#### public function beforeSave()

#### public function afterSave()

#### public function beforeDelete()

#### function setValueToAllRuleFields($fields, $value, $byPassEmpty = false)

Käime läbi kõik fieldsid, mis anname autoFillFields funktsioonist kaasa. Pärime objektist selle andmevälja väärtuse.

#### function validate()

Loome errors array. kui tekkis mingi viga valideerimisel siis tagastame false ehk kui $this->errors sees on mingi element, siis validate tagastab false, kui this->errors sees polnud midagi, siis tähendab polnud erroreid ja tagastatakse et validate õnnestus ehk true.

#### function resetAllAttributes()

Resetime kõik attribuudid default valudeks

#### function load($post)

$post sisu peab olema kujul ["fieldname" => $fieldValue, jne jne jne]. Ehk siis sobib nii form postist tulevnev info kui ka andmebaasist otse tulevnev vastus.

#### function delete()

Kustutab tabelist vastavad väärtused

#### function hasErrors()

Tagastab väärtuse juhul kui on tekkinud vigasid

#### function validateEmail($email)

Valideerib E-maili aadressi

### Detail View

#### static function widget($data)

Kuvab välja mudeli andmedväljad, mis on antud $data["attributes"] väärtuseks. Andmeväljade labelite nimetused tulevad mudeli klassist attributeLabels funktsioonist. Kui attributeLabels kandeid pole olemas, kuvatakse andmevälja enda nime alakriipsuteda ja suure algustähega.

### Flash

Hoiustatakse, lisatakse, eemaldatakse, päritakse ja kontrollitakse sõnumeid. Rakendatakse BaseModel-is valideerimisel tekkinud iga puhul, võib kasutada vabavoliliselt muul eesmärgil nt. Eduka sisselogimisel teadete saatmiseks.

#### function get()

Funktsioon tagastab flash objekti ja selle puudumisel luuakse Flash objekt sessioni sisse.

#### function setMessage($index, $message)

Päritakse Flash objekt ja määratakse funktsiooni sissetulevale sõnumile indeks.

#### function removeMessage($index)

Eemaldatakse indeksi kohal olev sõnum.

#### function getMessage($index)

Pärib sõnumi sisse tulnud sõnumist.

#### function hasMessage($index)

Kontrollib indeksis oleva sõnumi olemasolu.

#### function getMessages()

pärib kõik sõnumid Flash objekti sees.

#### function unsetMessages()

Määrab messagei uue tühja Array.

### GridView

#### function widget($data)

Kuvab välja mudeli andmedväljad, mis on antud $data["colums"] väärtuseks.

Andmeväljade labelite nimetused tulevad mudeli klassist attributeLabels funktsioonist. Kui attributeLabels kandeid pole olemas, kuvatakse andmevälja enda nime alakriipsuteda ja suure algustähega. Teeb View update „Delete“ nupu. ViewUpdate Delete nuppu saab konfida $data["buttons"] stringi kujul view/update/delete väärtuse.

### Helper

#### function generateRandomString($length = 128)

Genereerib randomiga stringi.

#### function getDegrees()

Määrab õppetaseme.

#### function generateSemesters($howManyYears = 3)

Genereerib millisel semetril aine toimub.

#### function sendMail($from, $to, $subject, $message)

Määrab aadressi kuhu vigade korral pöörduda tuleks

#### function setTitle($name)

Määrab lehe pealkirja.

#### function getTitle()

Pärib lehe pealkirja

#### function getClassName($obj)

Pärib klassinime

#### function isStringClean($str, $maxLength = 128)

Kontrollib sisestatud stringi puhtust.

#### function getCountries()

Määrab ära võimalikud riigid.

### Identity

Hoiab ja kontrollib kasutaja staatust ja sessioonivõtit.

#### function isLoggedIn()

Kontrollib kas kasutaja on sisselogitud või mitte, tagastab true/false väärtused.

#### function getUserId()

Tagastab sisselogitud kasutaja ID, selle puudumisel 0.

#### function isGuest()

Tagastab kas kasutaja on sisselogitud või mitte.

#### function validateIdentity()

Kontrillib kas kasutaja küpsise Auth\_Key vastab serveris olevale. Kui ei klapi logitakse kasutaja välja.

#### function login($user)

Määratakse identity objektile kõik vajalikud parameetrid.

#### function logout()

Eemaldatakse identity objektis sisselogitud kasutaja andmed.

#### function &get()

Staatiline funktsiooni millest tagastakse Identity objekt.

### QueryBuilder

Andmebaasiga suhtlemise selgroog.

#### function \_\_construct($tableName)

QueryBuilderi objekti loomisel antakse sisse tabeli nimi mis määratake QueryBuilderi objekti.

#### function select($tableName)

Luuakse queryBuilderi objekt kuhu antakse tableName querybuilderi objekti sisse ning määratakse päringu tüüp. Funktsioon tagastab QueryBuilderi objekti.

#### function addWhere($operator, $fieldName, $fieldValue)

Funktsiooni kasutakse päringute filtreerimiseks. Tagastab QueryBuilderi objekt.

#### function orWhere($operator, $fieldName, $fieldValue)

Funktsiooni kasutakse päringute filtreerimiseks. Tagastab QueryBuilderi objekt.

#### function refValues($arr)

funktsiooni kasutatakse stringi kaudu bindParam funktsiooni käivitamisel parameetrite ette andmiseks.

#### function composeParams($fieldValues)

Lisab binditud parameetride nimekirja andmetüüpide tähised.

#### function compose($limit = 0)

Koostatakse SQL päring stringi kujul vastavalt objekti omadustele.

#### function execute()

kävitatakse Compose funktsioon ja kasutab update, delete ja insert päringute puhul. Millel ei oodata vastust tagasi. Funktsioon tagastab endast ID väärtuse.

#### function query()

Funktsioon käivitab compose funktsiooni, mille järel käivitab päringu ja tagastab 1 andmebaasi kande, selle olemasolul.

#### function queryAll()

Funktsioon käivitab compose funktsiooni, mille järel käivitab päringu ja tagastab kõik andmebaasi kande, nende olemasolul.

#### function insert($tableName, $data)

Luuakse queryBuilderi objekt kuhu antakse tableName querybuilderi objekti sisse ning määratakse päringu tüüp. Funktsioon tagastab QueryBuilderi objekti. ['fieldname' => 'fieldvalue', jne]

#### function update($tableName, $data, $condition)

Luuakse queryBuilderi objekt kuhu antakse tableName querybuilderi objekti sisse ning määratakse päringu tüüp. Funktsioon tagastab QueryBuilderi objekti.

['fieldname' => 'fieldvalue', jne]

#### function delete($tableName, $condition)

Luuakse queryBuilderi objekt kuhu antakse tableName querybuilderi objekti sisse ning määratakse päringu tüüp. Funktsioon tagastab QueryBuilderi objekti.

["=", "fieldname", $fieldvalue]

### Request

#### function InitializeIfNotSet()

Käivitab parameetrite määramise kui neid pole eelnevalt määratud.

#### function setParams($params)

Määratab parameetrid.

#### function setCookie($name, $value)

Määrab küpsise väärtuse.

#### function setHeaders($headers)

Määrab päise.

#### function setUserAgent($userAgent)

Määrab kasutaja agendi.

#### function setUserIP()

Määrab kasutaja IP.

#### function setController($controller)

Määrab millist Controllerit hakatakse kasutama.

#### function setAction($action)

Määrab millist tegevust hakkab läbi viima.

#### function getParams()

Pärib määratud parameetreid

#### function getCookies()

Pärib määratud küpsist

#### function getHeaders()

Pärib määratud päiset

#### function getUserAgent()

Pärib määratud kasutaja agenti

#### function getUserIP()

Pärib määratud kasutaja IP

#### function getController()

Pärib milline controller läheb käiku

#### function getAction()

Pärib milline tegevus käivitub

#### function applyHeaders()

Kinnitab päised päringule

### Router

#### function parseUrl()

Parsitakse urli ja luuakse BaseControlleri objekt ning saadetakse URLi osad inital action funktsiooni.

## Config

### App

Sisaldab konfiguratsiooni e-mailile

### Db

Sisaldab konfiguratsiooni andmebaasi ühenduse loomiseks kujul:

host, user, pw(password), name

## Controllers

Järgnevad Controllerid omavad sarnast ülesehitust ja loogikat veebilehel informatsiooni kuvamiseks, lisamiseks, muutmiseks ja eemaldamiseks.

Seda nii vaate näol, kui ka AJAX-i päringute.

* CourseController
* CourseLearningOutcomeController
* CourseTeacherController
* DepartmentController
* SiteController
* SpecialityController
* StudyModuleController
* TeacherController
* TopicController
* CourseController
* UniversityController

### Action

Action mis tagastab render funktsiooni kuvab kasutajale funktsioonile etteantud vaate, tagastades JSON funktsiooni kaudu andmeid.

#### ActionIndex

Kuvab välja vastavas tabelis olevad kanded. Nimekirjal on viimases tulbas nupud, mis võimaldavad ühte tabeli kannet vaadata, muuta ja eemaldada. Nimekirja üleval on ühe kande loomise nupp.

#### actionCreate

Võimaldab lisada ühte andmebaasi tabelikannet.

#### actionUpdate

Võimaldab muuta ühte andmebaasi tabelikannet.

#### actionView

Võimaldab vaadata ühte andmebaasi tabelikannet.

#### actionSave

Võimaldab lisada ja muuta üte andmebaasi kannet AJAX-i kaudu.

#### actionRemove

Võimaldab eemaldada ühte andmebaasi tabelikannet AJAX-i kaudu.

#### actionGet\*Subelement\*(sõltub tabelist)

Tagastab JSON Array alamelemendidega ja staatuse ning veateatega, selle olemasolul.

#### findModel

Pärib andmebaasist ühe tabeli kande info ja lisab modeli, mille sisse lisatakse päringu tulemus. Tagastatakse vastava modeli väärtus.

### UserController

Omab peale teiste controllerite võimekust veel kasutaja sisse- ja väljalogimist.

## Models

Vastavad Model-id kirjeldavad neile vastavaid andmebaasi tabeleid ja omavad automaat täitmise võimekust.

Erandideks on Form ja SystemLog tüüpi Model-id. Forme kasutatakse POST päringutel andmete kätte saamiseks. Log-i kasutatakse vaid rakenduses logimise eesmärgil.

* ForgotPasswordForm
* LoginForm
* PartnerUniversitySystemLog

## Web

### .htaccess

Teeb võimalikus Controllerite ja actionite URL-ide töötlemise.

### Autoloader

Kõik failid mida kasutatakse rakenduse töö tagamiseks. Sealne failide sisselaadimise järjekord on oluline, kuna failid mida kasutakse ennem, peavad olema laetud eelnevalt, kui muud failid. Teeb võimalikus namespace-ide ja erinevate classide kättesaadavuse. Ei sisalda view faile.

### Index.php

Esimene fail millest kogu rakenduse töö algab.

### Main.js

Omab üldlehe tarbeks vajaminevaid funktsionaalsuseid.