* \_template
  + footer.php
  + header.php
* app
  + controller
    - LoginController.php
      * public static void doLogin()
      * public static void doLogout()
      * public static String getLoginWelcome()
    - RegistrationController.php
      * public static void doRegistration()
      * public static User getUserFromPost()
    - RequestController.php
      * public static String getEscapedHTMLPost($post\_key)
      * public static String getCleanPost ($post\_key)
      * private static String escapeHTML ($string)
      * private static String clean ($string)
  + core
    - Config.php (private static $configuration, $environment)
      * public static void \_\_init($environment)
      * public static String get ($config\_key)
      * private static void autoloadClass($class\_name)
      * private static String getEnv()
    - Feedback.php
      * public static void add($category, $feedback)
      * public static String getFeedbackList ($category,)
      * public static String getAllFeedbackList ()
      * public static String wrapFeedbackList ($list)
    - Query.php (private $tableName,$select,$where,$limit,$insert,$update)

(private $db,$sql,$query,$connection,$isExecSuccess)

* + - * public void \_\_construct($table\_name)
      * public void selectCols($col\_names = array())
      * public void updateCols ($col\_names)
      * public void whereColsEqual ($col\_names)
      * public void limit $limit\_num)
      * public void insert ($col\_names)
      * public void prepare($bind\_names, $bind\_values)
      * public void execute()
      * public bool isSuccess()
      * public Object getResult()
      * public bool isFieldValueTaken($field\_names, $field\_values)
    - QueryFactory.php
      * public static String buildSelect($table, $col\_names)
      * public static String buildWhere($col\_names, $comparison\_ops,

$logical\_op = 'AND')

* + - * public static String buildLimit($limit\_num)
      * public static String buildInsert($table, $col\_names)
      * public static String buildUpdate($table, $col\_names)
      * private static String buildSeparatedString($list\_items, $delimeter)
      * private static bool areSameSize($list\_of\_arrays)
    - SQLiteDB.php (private $path,$name)
      * public void \_\_construct($name = null, $path = null)
      * public PDO getNewConnection()
      * private static void createTables($connection)
    - User.php (public $username,$email,$fullName)
      * public void \_\_construct($username, $email)
      * public bool isValidSyntax($show\_feedback = false)
      * public bool isValidUsernameSyntax($show\_feedback = false)
      * public bool isValidEmailSyntax($show\_feedback = false)
      * public bool isUserTaken()
      * public bool isEmailTaken()
      * public bool isUsernameTaken ()
      * public bool isValidPassword($password)
      * public bool addToDB($password)
      * public void loadUserFromDB()
  + model
    - LoginModel.php
      * public static bool isUserLoginValid($user)
      * public static bool isUserPasswordValid($user, $password)
      * private static bool isValidCredentials($login\_type, $login\_value,

$password)

* + - * private static String getLoginType($user)
      * public static void setLoginStatus($status, $user = "")
      * public static bool isLoggedInAs($user)
      * public static String getLoginStatus()
      * private static void setLoginTimestamp($user)
    - RegistrationModel.php
      * public static bool addNewUser($user, $password)
  + view
    - IndexView.php
    - LoginView.php
    - RegistrationView.php
    - View.php
* css
  + style.css
* js
  + jquery.js
  + script.js
* config.development.php
* index.php
* .db