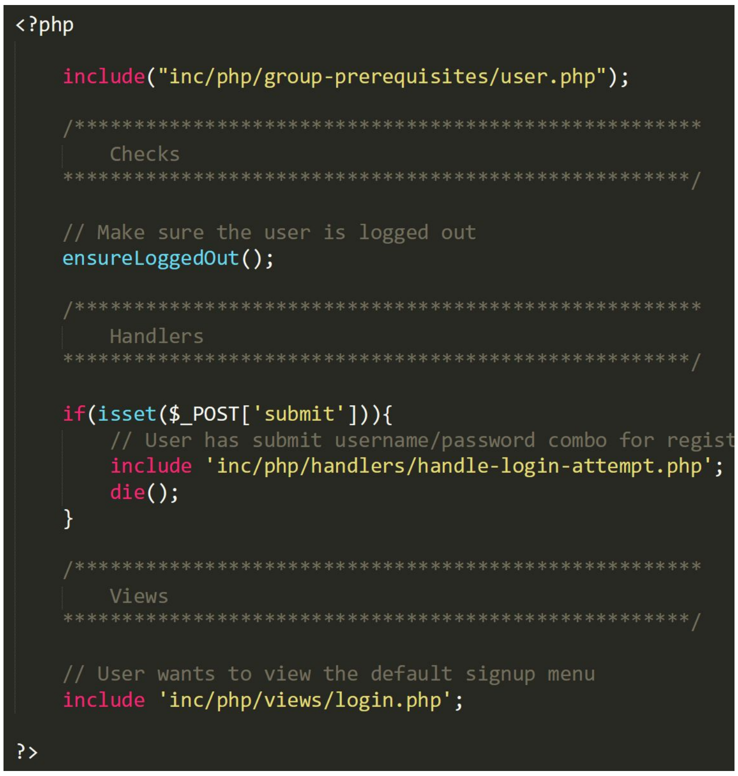
Account System
Technical Specification

Structure

The system is based very loosely on the MVC design pattern. Each top-level file (sibling of index) has three sections: **checks**, **handlers** and **views**. Benefits of using this format include: increased readability, and the ability to set a form's action attribute to '?' (redirecting to the same file once submitted, keeping relevant code together).



```php
<?php

    include("inc/php/group-prerequisites/user.php");

    /*****************************************************
        Checks
    *****************************************************/

    // Make sure the user is logged out
    ensureLoggedOut();

    /*****************************************************
        Handlers
    *****************************************************/

    if(isset($_POST['submit'])){
        // User has submit username/password combo for regist
        include 'inc/php/handlers/handle-login-attempt.php';
        die();
    }

    /*****************************************************
        Views
    *****************************************************/

    // User wants to view the default signup menu
    include 'inc/php/views/login.php';

?>
```

The image shown to the left (login.php) provides a simple example of how this works.

Checks are anything which must be run before the user is sent header information. In login.php, we're checking if the user is logged out.

Note: 'group-prerequisites/user' contains some functions which can be applied across multiple pages. The function 'ensureLoggedOut()' redirects the user to index if it detects that session variables have already been set for the user.

The **handlers** section is where, based on if a form has been submitted, a file is included to process the sent information. In login.php, the login attempt handler has been included to process the user's account credentials.

If the code is still running by the time it reaches the **views** section, it's because the user is allowed to be here, and needs to view the page. The file 'login.php' contains code to display a login form.

Validation Regex

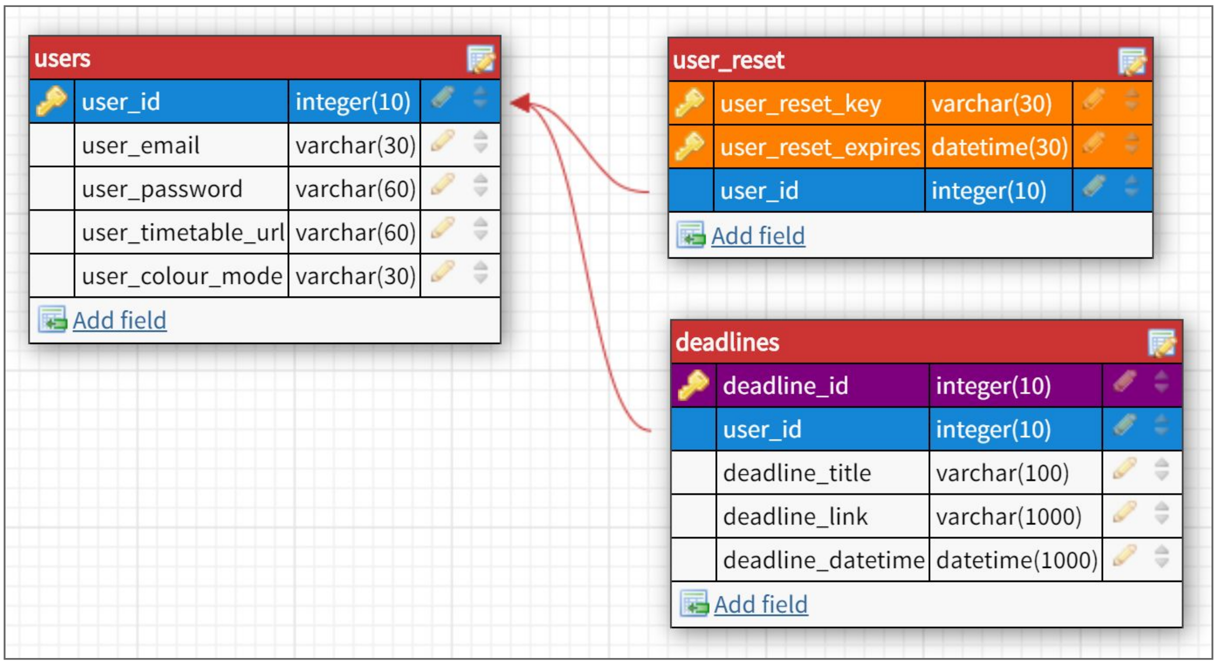
Email
```
/^([a-zA-Z0-9_\-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(
([a-zA-Z0-9\-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$/
```

Password
```
/^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])[0-9a-zA-Z]{8,}$/
```

Note: On signup, an email is considered invalid if it is already associated with an account.

Database

Below is an image showing only the tables associated with the account system.



users

🔑	user_id	integer(10)
	user_email	varchar(30)
	user_password	varchar(60)
	user_timetable_url	varchar(60)
	user_colour_mode	varchar(30)

Add field

user_reset

🔑	user_reset_key	varchar(30)
🔑	user_reset_expires	datetime(30)
	user_id	integer(10)

Add field

deadlines

🔑	deadline_id	integer(10)
	user_id	integer(10)
	deadline_title	varchar(100)
	deadline_link	varchar(1000)
	deadline_datetime	datetime(1000)

Add field

Users Table

The users table is used to store login credentials for users, as well as some personalization related data.

- The user's password is hashed before being entered into the table.
- "user_colour_mode" is just a keyword which corresponds to a specific CSS file.
- "User_timetable_url" is the only attribute which can be null.

User Reset Table

The user reset table is used to store a key and expiry date for a password change.

- No attribute in this table can be null.
- When a user deletes their account, any associated entries are deleted from this table.

Note: When a key expires, it doesn't automatically get deleted - a user must try to use it before it's removed from the database. A potential option for the future is to have a PHP script run once monthly to remove all expired entries.

Deadlines Table

The deadlines table is used to store a user's deadline entries.

- No attribute in this table can be null.
- When a user deletes their account, all of their deadlines are deleted too.

Languages

- HTML, CSS - Used throughout the account system.
- JavaScript - Used for form validation.
- PHP - Used for server-side logic.
- SQL - Used to store user, password reset and deadline data.