

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| **Document title** | **CollabTogetherJS** |
| **Document reference number** | **UEL -** |
| **Project name** | **UNISA e-Content 20140363** |
| **Unisa Department** | **Department of Economics** |

Table of Contents

1 Description Of Component 5

2 How to use the Component 5

3 How to use the Component 7

Document sign-off

This document has been seen and accepted by the following people:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Title | Date | Signature |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Compiler | Purpose | Change Summary |
| <yyyymmdd> | Name Surname | Document Created,  Internal Review,  External Review,  etc | Initial release,  Updates based on internal review,  Updates to section xxx,  etc |
|  |  |  |  |
|  |  |  |  |

# Description of the component

COllabTogether is a component which allows student to collaborate on an assignment, a task, or lecturer can give instructions to students.

# How to install CollabTogether Compoenent

## Installation:

## Step 2.2

Install Laravel framework. The following command will install the latest version of Laravel with the project name LaravelDefault.

composer create-project laravel/laravel LaravelDefault

## Step 2.3

Download Zip from [CollabTogetherJS Git repo](https://github.com/EonConsulting/CollabTogetherJS).

## Step 2.4

Create a folder called "Packages" inside of the root of the Laravel project.

## Step 2.5

Paste the folder from the Git repo (Step 1.2) inside the packages folder.

## Step 2.6

Inside the root Laravel project, open the composer.json file. In that file, look for:

"autoload" : {

"psr-4": {

...

Inside of the psr-4 tag, load the following:

"EONConsulting\\Collab\\": "packages/CollabTogetherJS/src",

## Step 2.7

Use as a global function instead of by namespace (Optional)

Under the psr-4 but still inside of the autoload if the array files does not exist, create it like this:

"files": [

...

]

And enter into it:

"packages/CollabTogetherJS/src/Http/helpers.php",

## Step 2.8 Registering the Service Provider

Open up config/app.php. In the providers array, enter the following:

\EONConsulting\Collab\CollabServiceProvider::class,

## Step 2.9

In the same file, in the aliases array, enter:

'Collab' => \EONConsulting\Collab\src\Facades\Collab::class,

## Step 2.10

To be able to use the tsugi, the APP\_URL in your env file needs to be correct. So change it to the URL you are using for your testing environment.

## Step 2.11

In the command line / terminal, enter the following command in the root of the Laravel project:

composer dump-autoload -o

php artisan vendor:publish

## Step 2.12 Testing

Let's see this in action. Go to routes/web.php and where it says return view('welcome');replace that with

return Collab::collaborate();

// or

return collab()->collaborate(); // if you did step 2.4

To test this out, go to the URL for this Laravel site.

You should now see the "Start Collaboration" button being printed out to the screen.

## Step 2.13 Tsugi Config

We have provided a very easy way to use Tsugi.

You just need to setup the keys in your .env file. Like so:

TSUGI\_DB=tsugi

TSUGI\_DB\_HOST=127.0.0.1

TSUGI\_DB\_USER=root

TSUGI\_DB\_PASS=root

TSUGI\_DB\_PREFIX=""

TSUGI\_ADMIN\_PASS=1234

TSUGI\_SERVICE\_NAME="TSUGI"

TSUGI\_SERVICE\_DESC=""

TSUGI\_OWNER\_NAME="Charles Severance"

TSUGI\_OWNER\_MAIL=""

TSUGI\_PROVIDE\_KEYS=false

TSUGI\_GOOGLE\_CLIENT\_ID=false

TSUGI\_GOOGLE\_CLIENT\_SECRET=false

TSUGI\_GOOGLE\_MAP\_API\_KEY=false

TSUGI\_BADGE\_ENCRYPT\_PASSWORD=false

TUSIG\_BADGE\_ASSERT\_SALT=false

TSUGI\_PRODUCT\_INSTANCE\_GUID="lti2.example.com"

TSUGI\_DEVELOPER=true

TSUGI\_COOKIE\_SECRET="warning:please-change-cookie-secret-a289b543"

TSUGI\_COOKIE\_NAME="TSUGIAUTO"

TSUGI\_COOKIE\_PAD="390b246ea9"

TSUGI\_MAIL\_DOMAIN=false

TSUGI\_MAIL\_SECRET="warning:please-change-mailsecret-92ds29"

TSUGI\_MAILEOL="\n"

TSUGI\_NONCE\_CHECK=100

TSUGI\_NONCE\_TIME=1800

TSUGI\_SESSION\_SALT="warning:please-change-sessionsalt-89b543"

TSUGI\_TIMEZONE="Pacific/Honolulu"

TSUGI\_OLD\_ANALYTICS\_KEY=false

TSUGI\_OLD\_ANALYTICS\_NAME=false

TSUGI\_UNIVERSAL\_ANALYTICS=false

TSUGI\_OFFLINE=false

Fill out these config details to best suit your project. If you do not know what the key means or what to change it to, leave it to its default value above.

# How to use the CollabTogetherJS

For the demo, we have included 3 different views to show how this tool works.

Clear / Empty page

A contact form

A multi-level menu

## Clear / Empty page

To get to this view, simply call:

return collab()->collaborate();

## A contact form

To get to this view:

return collab()->demo\_contact\_form();

## A multi-level menu

To get to this view:

return collab()->demo\_menu();

## Add the Collaboration Tool to any page

Copy the button to desired location.

<button id="btn\_collab" onclick="TogetherJS(this); return false;">Start Collaboration</button>

*Note:* You do not have to use a button, you can use any element, but to get the tool to start, you need to call the TogetherJS(this); function in JavaScript.

## Scripts

Include these scripts at the bottom of your page (just above the body tag.

<script

src="https://code.jquery.com/jquery-2.2.4.min.js"

integrity="sha256-BbhdlvQf/xTY9gja0Dq3HiwQF8LaCRTXxZKRutelT44="

crossorigin="anonymous"></script>

<script src="https://togetherjs.com/togetherjs-min.js"></script>

<script>

$(function () {

var visibilityChangeFromRemote = false;

var MyApp = TogetherJS;

$("#start-togetherjs").on('click', function() {

TogetherJS(this);

});

TogetherJSConfig\_on = {

ready: function () {}

};

TogetherJSConfig\_on\_ready = function () {

MyApp.on("visibilityChange", fireTogetherJSVisibility);

console.log('on ready');

};

TogetherJSConfig\_on\_close = function () {

MyApp.off("visibilityChange", fireTogetherJSVisibility);

};

function fireTogetherJSVisibility(element, isVisible) {

if (visibilityChangeFromRemote) {

return;

}

var elementFinder = TogetherJS.require("elementFinder");

var location = elementFinder.elementLocation(element);

TogetherJS.send({type: "visibilityChange", isVisible: isVisible, element: location});

}

TogetherJS.hub.on("visibilityChange", function (msg) {

var elementFinder = TogetherJS.require("elementFinder");

// If the element can't be found this will throw an exception:

var element = elementFinder.findElement(msg.element);

MyApp.changeVisibility(element, msg.isVisible);

});

TogetherJS.hub.on("togetherjs.init-connection", function (msg) {

//var session = TogetherJS.require("session");

//var peers = TogetherJS.require('peers');

console.log('session', session);

console.log('peers', peers.getAllPeers());

$('#btn\_collab').html('Stop Collaboration');

});

TogetherJS.hub.on("togetherjs.hello", function (msg) {

if (! msg.sameUrl) {

return;

}

MyApp.allToggleElements.forEach(function (el) {

var isVisible = $(el).is(":visible");

fireTogetherJSVisibility(el, isVisible);

});

});

TogetherJS.hub.on("togetherjs.bye", function (msg) {

var session = TogetherJS.require("session");

console.log('session', session);

$('#btn\_collab').html('Start Collaboration');

});

$.fn.syncShow = function () {

this.show();

this.trigger("visibilityChange");

};

$.fn.syncHide = function () {

this.hide();

this.trigger("visibilityChange");

};

$(document).on("visibilityChange", function () {

MyApp.emit("visibilityChange", this, $(this).is(":visible"));

});

MyApp.changeVisibility = function (el, isVisible) {

if (isVisible && ! el.is(":visible")) {

el.syncShow();

} else if ((! isVisible) && el.is(":visible")) {

el.syncHide();

}

};

});

</script>

Appendix A

For Appendices use the style “UEL Appendix”