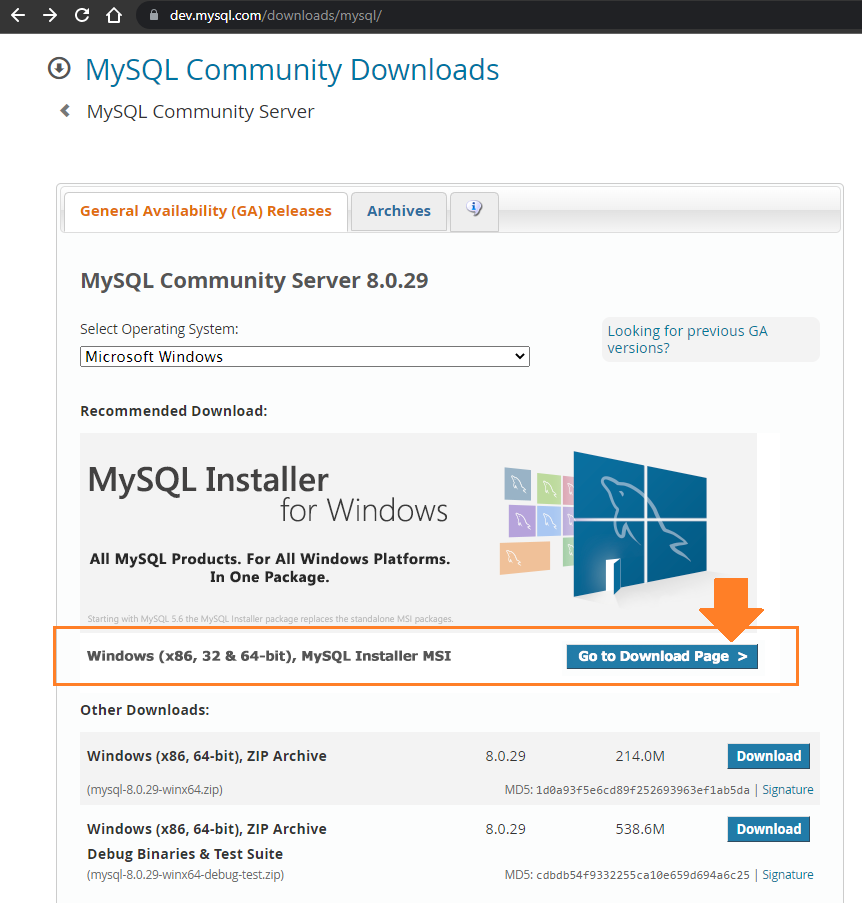
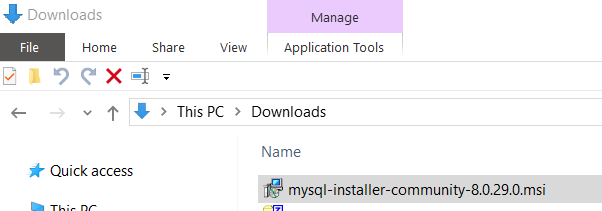
# REACH Chat Local Setup Procedure

## Install MySQL

* 1. Go to <https://dev.mysql.com/downloads/mysql/>
  2. Download MySQL Community Server version 8.

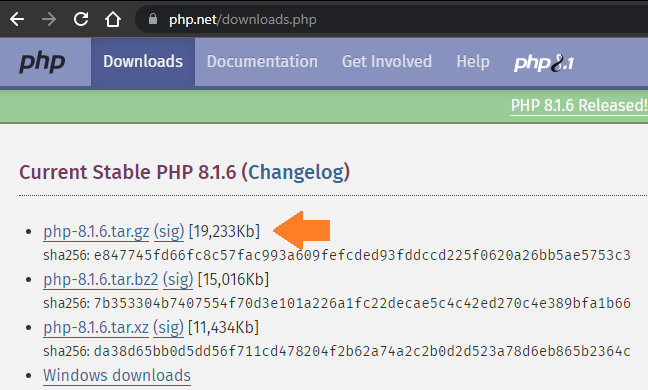


* 1. Once downloaded, install MySQL Server on the local computer

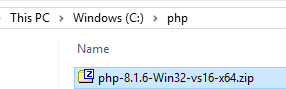


## Install PHP

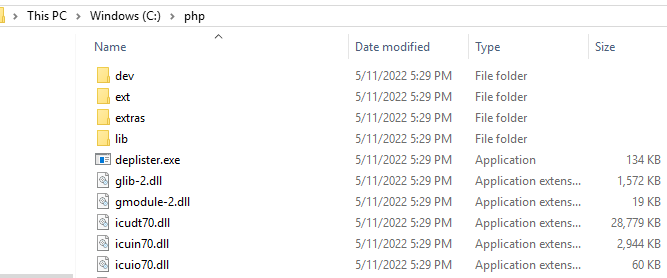
1. Go to <https://www.php.net/downloads.php>
2. Download the current stable PHP version



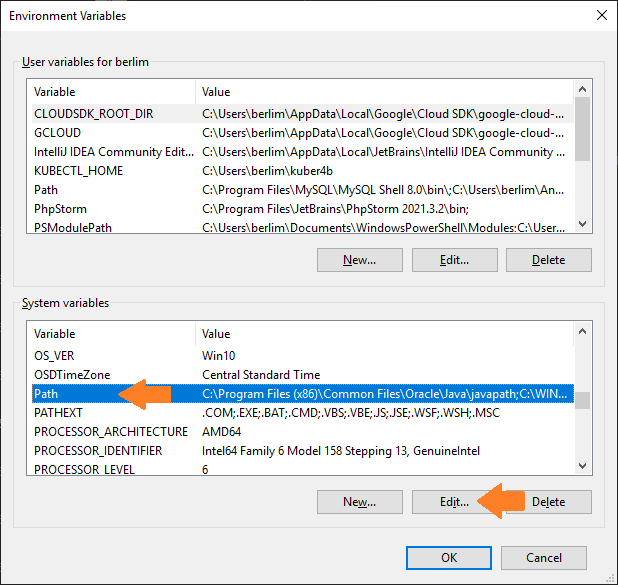
1. Create a new folder called “php” under C: drive
2. Move or copy the downloaded file to C:\php folder

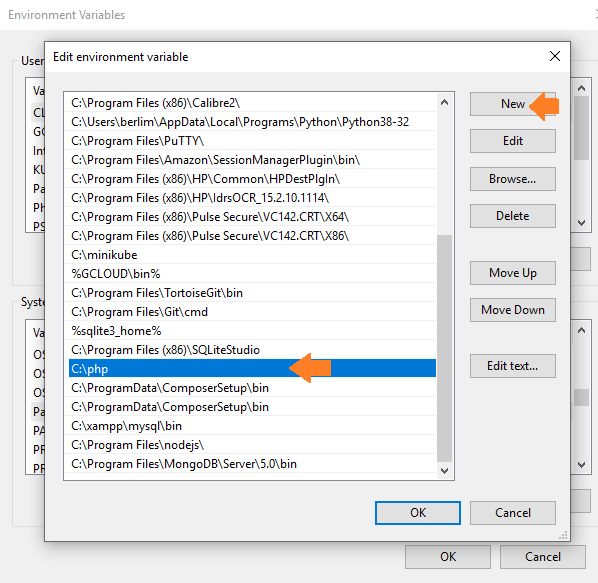


1. Extract the zip file



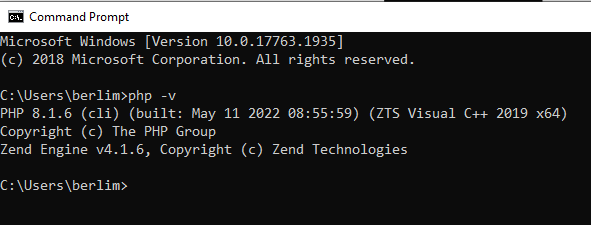
1. Add the PHP path to Windows System Environment Variables.





1. Open Command Prompt and check php version using php -v command. In this example, PHP version 8.1.6 was installed.

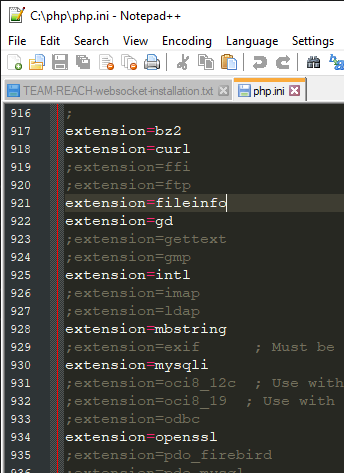
php -v



## Enable PHP extensions

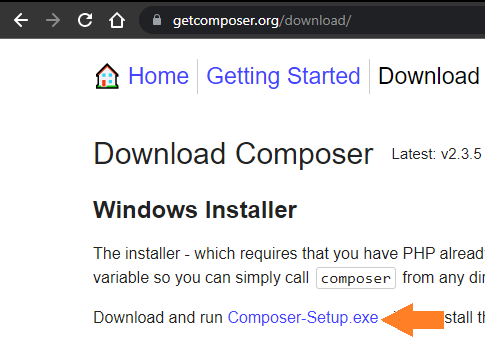
1. Edit C:\php\php.ini and enable PHP extensions by uncommenting the lines for the following extensions:

bz2, curl, fileinfo, gd, intl, mbstring, mysqli, openssl, pdo\_mysql

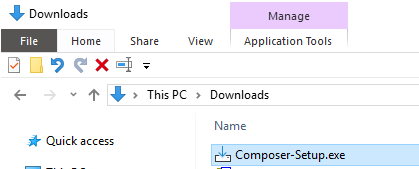


## Install composer

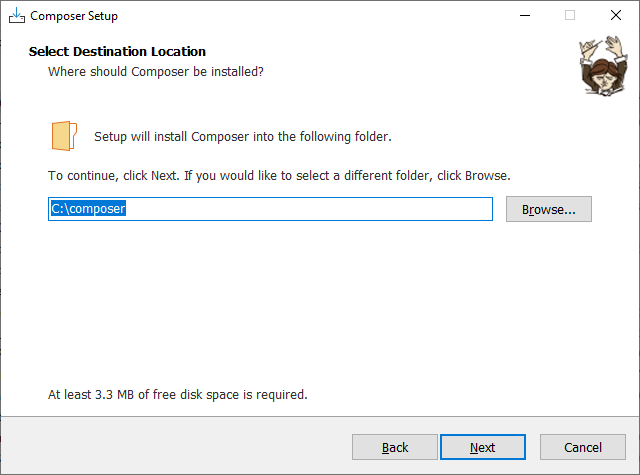
1. Go to <https://getcomposer.org/download/> to download the composer installer.



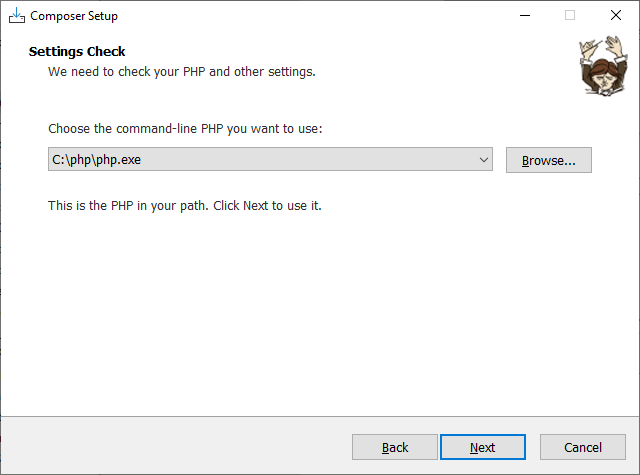
1. Double-click “Composer-Setup.exe” to install composer.



1. Select C:\composer as the target directory.



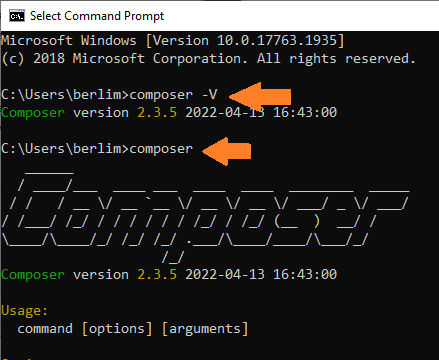
1. Select the location of the previously installed PHP. In my case, I installed PHP at C:\php directory.



1. Once composer is installed, open Command Prompt to verify composer.

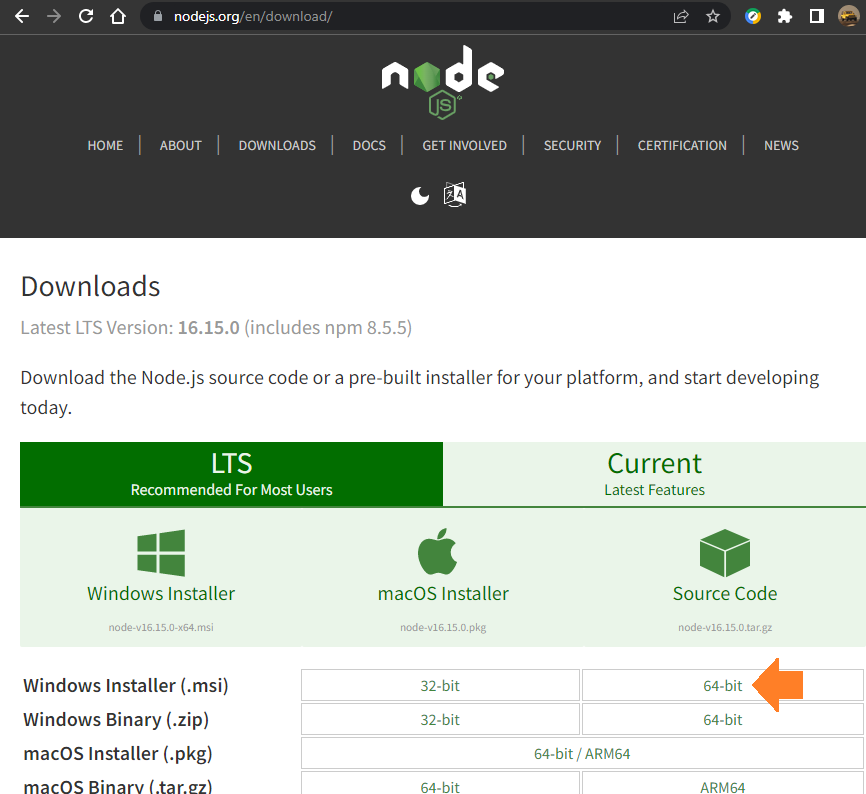
composer -V

composer

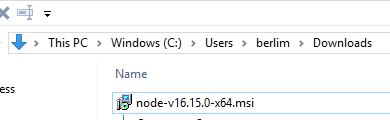


## Install NodeJS

1. Go to <https://nodejs.org/en/download/> to download the NodeJS installer.



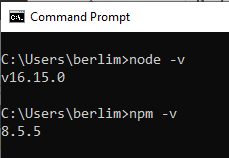
1. Execute the installer file.



1. Once installed, open Command prompt and check the Nodejs version.

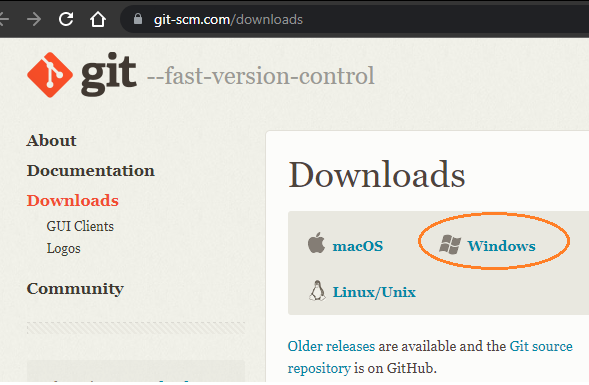
node -v

npm -v



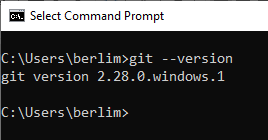
## Install Git

1. Go to <https://git-scm.com/downloads> to download the Git installer.



1. Install Git
2. Once installed, check the Git version using command prompt

git --version



## Clone the REACH chat application from GitHub

1. Go to the target directory where you want to put the project code that will be cloned from GitHub. In this example, the target directory is GitHub under Desktop directory.

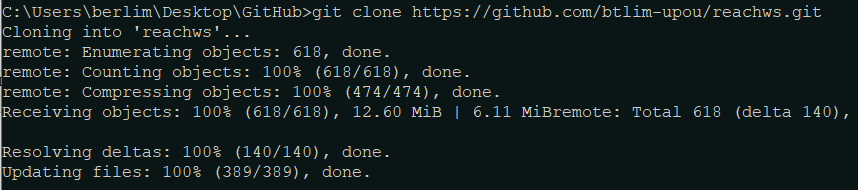
C:\Users\berlim\Desktop>mkdir GitHub

C:\Users\berlim\Desktop>cd GitHub

C:\Users\berlim\Desktop\GitHub>

1. Clone the project from GitHub.

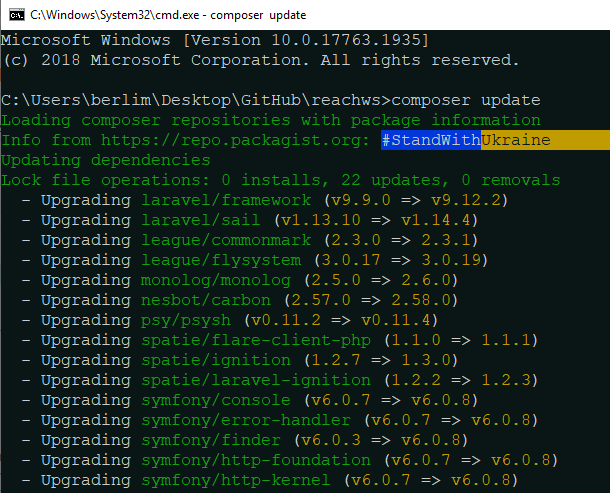
C:\Users\berlim\Desktop\GitHub>git clone https://github.com/btlim-upou/reachws.git



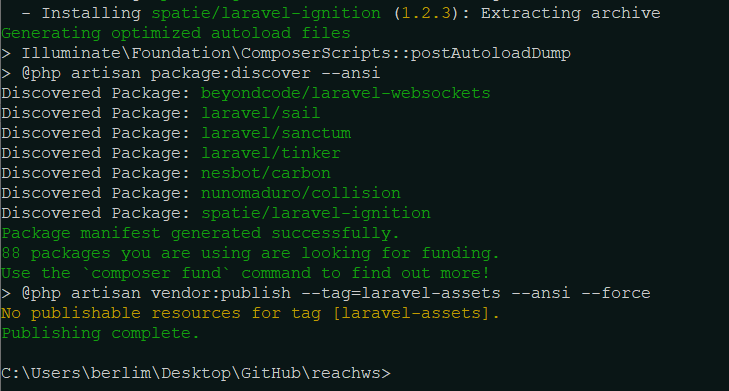
## Run Composer Update

1. In the project directory, run “composer update”.

C:\Users\berlim\Desktop\GitHub\reachws>composer update



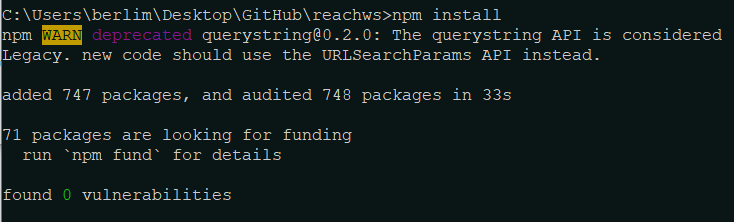
1. The installation will take several minutes to complete. You should see a similar output below that confirm a successful installation.



## Run npm install

1. In the project directory, run “npm install”.

C:\Users\berlim\Desktop\GitHub\reachws> npm install



## Create MySQL database

1. In the project directory, login to mysql server.

C:\Users\berlim\Desktop\GitHub\reachws>mysql -u root -p

Enter password: \*\*\*\*\*\*\*\*\*\*\*\*\*

*Welcome to the MariaDB monitor. Commands end with ; or \g.*

*Your MySQL connection id is 8*

*Server version: 8.0.28 MySQL Community Server - GPL*

*Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.*

*Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.*

MySQL [(none)]> show databases;

1. Create the MySQL database by executing the reachdb.sql file.

MySQL [(none)]> source reachdb.sql

*Query OK, 1 row affected (0.012 sec)*

*Database changed*

*Query OK, 0 rows affected (0.001 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*Query OK, 0 rows affected, 1 warning (0.000 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*Query OK, 0 rows affected (0.000 sec)*

*MySQL [reachdb]>*

1. Create the MySQL user and grant permissions to reachdb database.

MySQL [reachdb]> create user 'rchuser'@'localhost' identified by "uxdMMv4332";

*Query OK, 0 rows affected (0.014 sec)*

MySQL [reachdb]> grant all privileges on \*.\* to 'rchuser'@'localhost';

*Query OK, 0 rows affected (0.013 sec)*

MySQL [reachdb]> flush privileges;

*Query OK, 0 rows affected (0.013 sec)*

## Create and edit .env file

1. Create a new .env file in the project directory. You can copy the .env.example file to create a new .env file.
2. Edit the .env file with correct DB connection information.

APP\_URL=http://localhost:8000

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=reachdb

DB\_USERNAME=rchuser

DB\_PASSWORD=uxdMMv4332

BROADCAST\_DRIVER=pusher

MAIL\_MAILER=smtp

MAIL\_HOST=smtp.googlemail.com

MAIL\_PORT=465

MAIL\_USERNAME=reachat207@gmail.com

MAIL\_PASSWORD=R3@ch207!

MAIL\_ENCRYPTION=ssl

MAIL\_FROM\_ADDRESS="reachat207@gmail.com"

MAIL\_FROM\_NAME="${APP\_NAME}"

PUSHER\_APP\_ID=anyID

PUSHER\_APP\_KEY=anyKey

PUSHER\_APP\_SECRET=anySecret

PUSHER\_APP\_CLUSTER=mt1

## Edit broadcasting.php and bootstrap.js files to connect to a remote websocket server

**Note**:

We will be starting this application locally using “php artisan serve”, which does not support TLS or HTTPS. Therefore, this application will not be able to connect to a remote websocket server that was already configured with TLS/SSL (e.g., ws.allhumans.one).

We will be starting a local websocket server using php artisan websockets:serve command to be used for local testing.

1. Edit **config/broadcasting.php**.

'pusher' => [

'driver' => 'pusher',

'key' => env('PUSHER\_APP\_KEY'),

'secret' => env('PUSHER\_APP\_SECRET'),

'app\_id' => env('PUSHER\_APP\_ID'),

'options' => [

'cluster' => env('PUSHER\_APP\_CLUSTER'),

'useTLS' => false,

'encrypted' => true,

'host' => 'localhost',

'port' => 6001,

'scheme' => 'http'

],

1. Edit **resources/js/bootstrap.js** to connect to the remote websocket server at ws.allhumans.one.

window.Echo = new Echo({

broadcaster: 'pusher',

key: process.env.MIX\_PUSHER\_APP\_KEY,

// cluster: process.env.MIX\_PUSHER\_APP\_CLUSTER,

forceTLS: false,

encrypted: true,

wsHost: 'localhost',

wsPort: 6001,

// wssPort: 6001,

disableStats: true

});

1. In the project directory, run “**npm run dev**” to compile the changes to public app.js file.

C:\Users\berlim\Desktop\GitHub\reachws>npm run dev

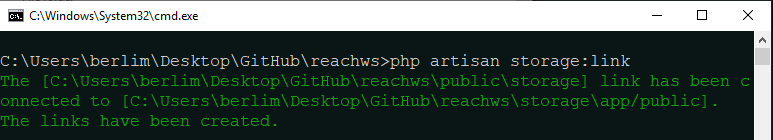
## Create the directory for attachments

1. In the project directory, link the public storage directory using “**php artisan storage:link**” commad.

C:\Users\berlim\Desktop\GitHub\reachws> php artisan storage:link

*The [C:\Users\berlim\Desktop\GitHub\reachws\public\storage] link has been connected to [C:\Users\berlim\Desktop\GitHub\reachws\storage\app/public].*

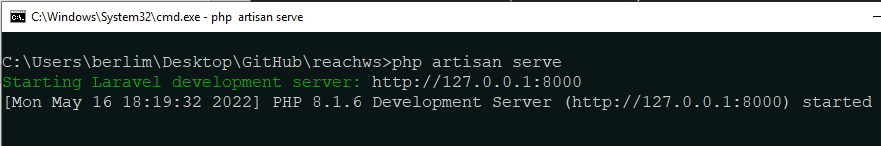
*The links have been created.*



1. Create the **attachments** directory under **storage/app/public** directory.

C:\Users\berlim\Desktop\GitHub\reachws>cd storage/app/public/

C:\Users\berlim\Desktop\GitHub\reachws\storage\app\public>mkdir attachments



## Generate a new key

1. In the project directory, generate a new key

C:\Users\berlim\Desktop\GitHub\reachws> php artisan key:generate

*Application key set successfully.*

## Start a local websocket server

**Note**:

We will be starting this application locally using “php artisan serve”, which does not support TLS or HTTPS. Therefore, this application will not be able to connect to a remote websocket server that was already configured with TLS/SSL (e.g., ws.allhumans.one).

We will be starting a local websocket server using php artisan websockets:serve command to be used for local testing.

1. Run **php artisan websockets:serve** to start the websocket server.

C:\Users\berlim\Desktop\GitHub\reachws>php artisan websockets:serve

*Starting the WebSocket server on port 6001...*

## Start the REACH application using “php artisan serve”

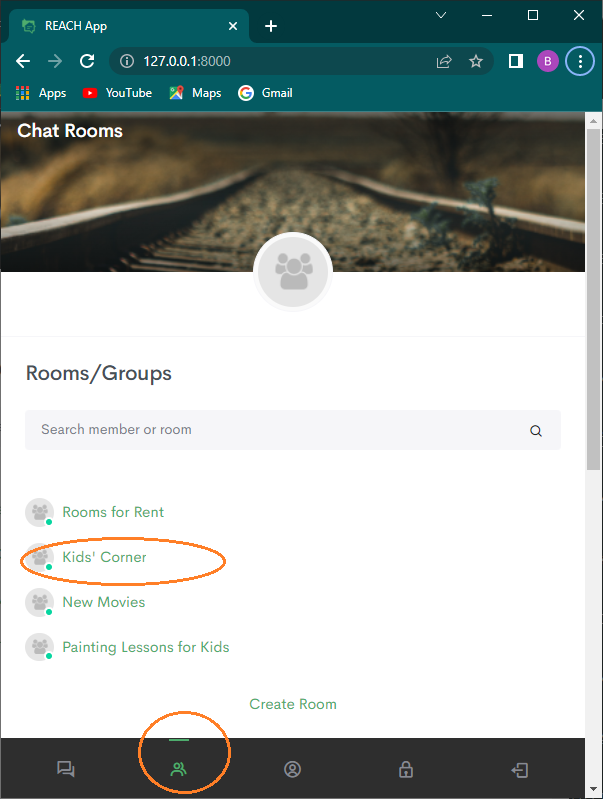
1. Run **php artisan serve** to start the application.

C:\Users\berlim\Desktop\GitHub\reachws> php artisan server

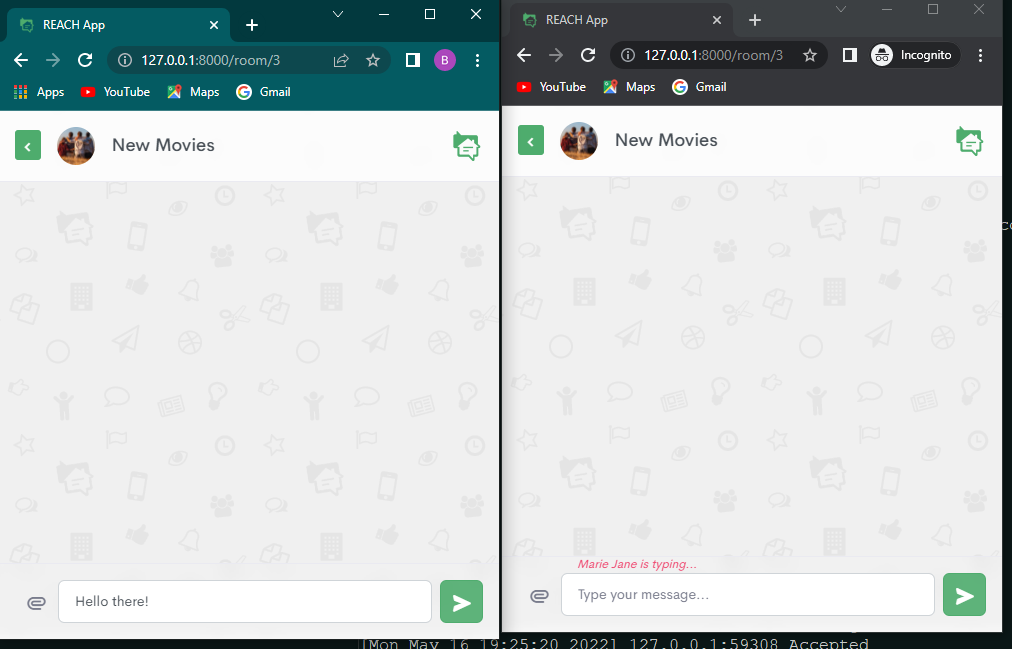
1. In the browser, go to <http://127.0.0.1:8000/>
2. Register a new user or login using one of the existing users:

|  |  |
| --- | --- |
| Username | Password |
| qhill@example.org | password |
| ireichert@example.org | Password |
|  |  |

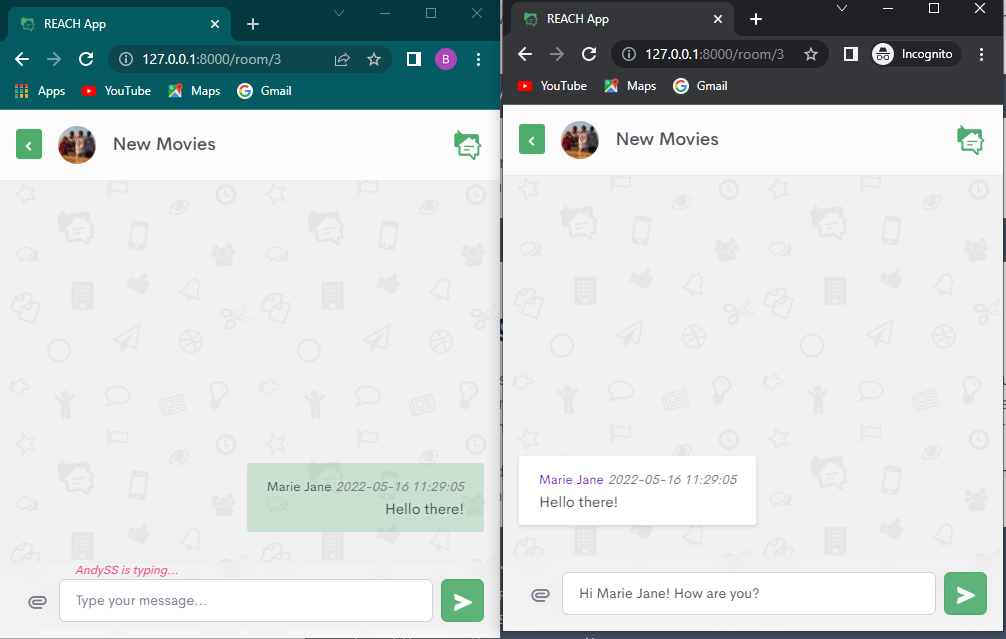
1. Click the Rooms tab, and select a room.

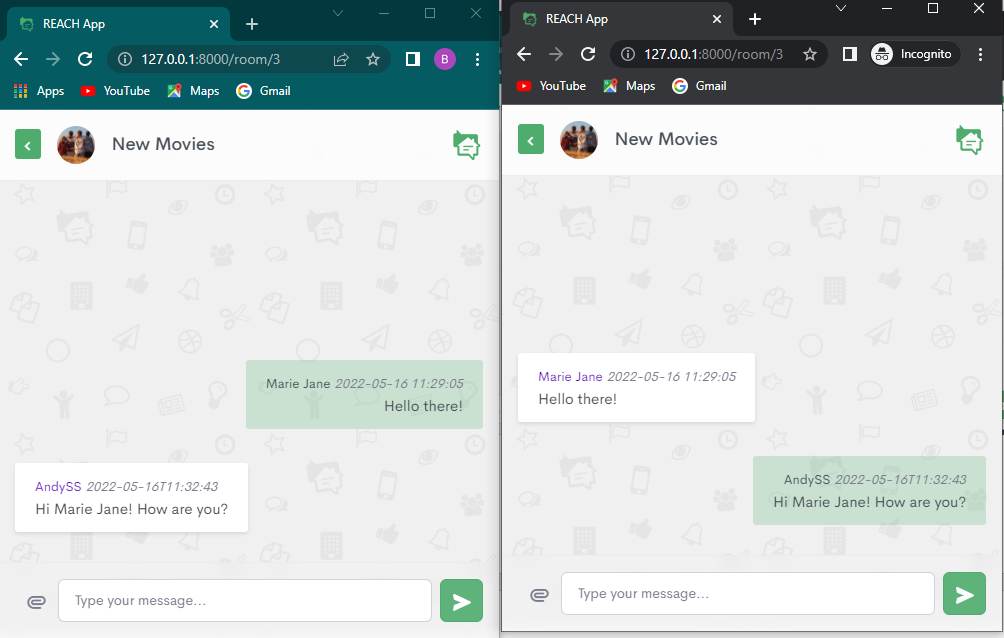


1. Open another browser window in private or incognito mode. Go to <http://127.0.0.1:8000/> again and login as another username to simulate the chat between 2 users.
2. In the second session, select the same room.
3. Type a message in one session. The *typing status* should show in the other session.

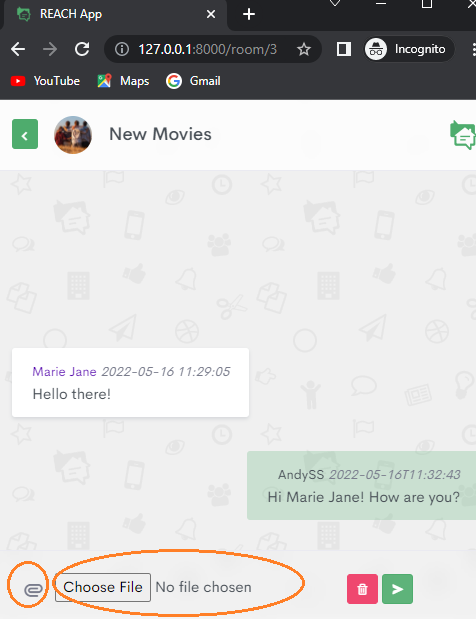
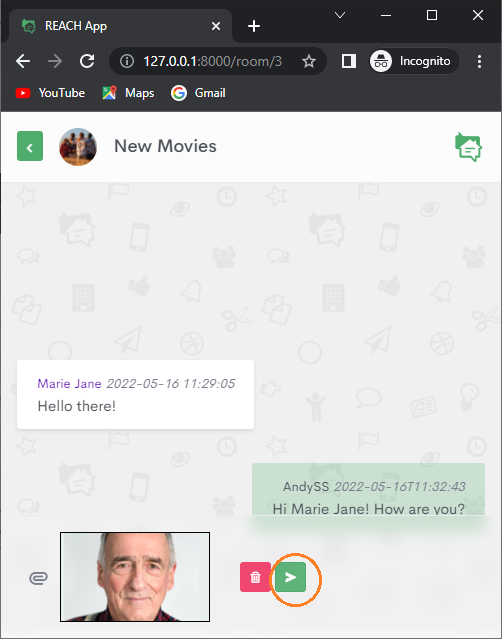


1. Respond in the other session.

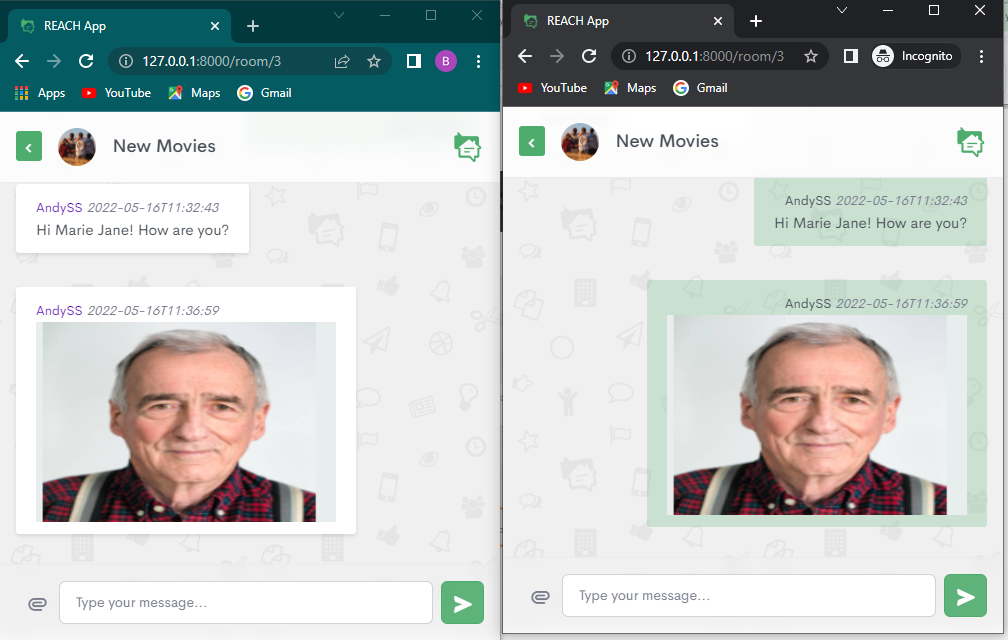




1. Click the “Attach” button and select file. For now, only images can be sent. The selected image appears. Click send.

1. The attachment appears on both sessions.



End of document