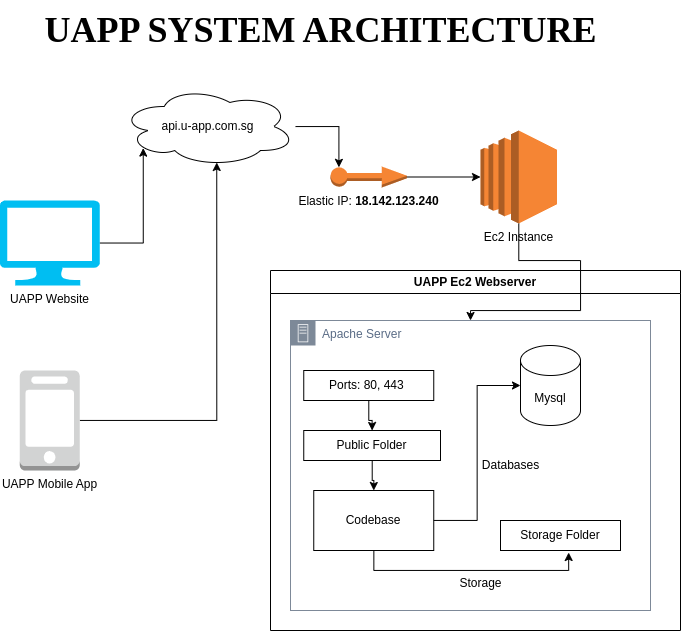
**UAPP Documentation (External)**

# **Introduction**

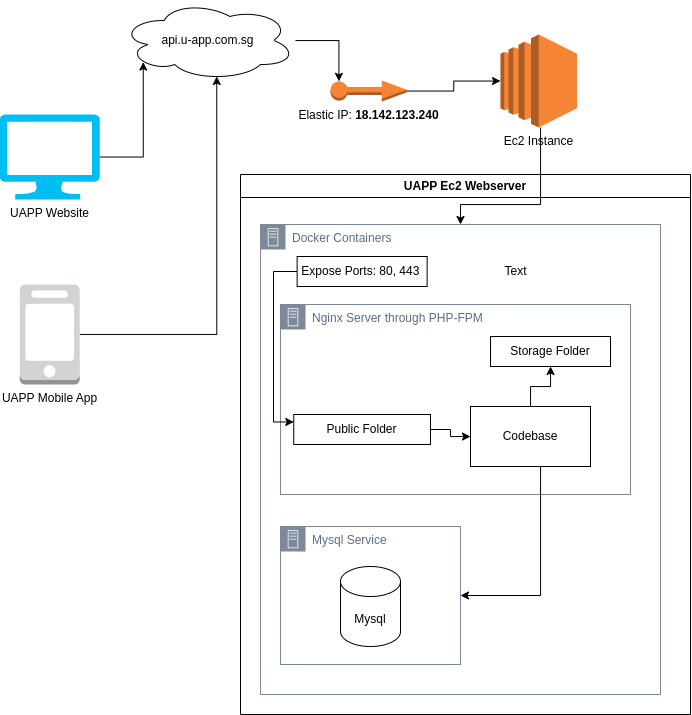
The dynamic system comprises a website and a mobile app. The website functions as the backend access for administrators and serves as the web server for the mobile app. Meanwhile, the mobile app serves as the primary interface for users to interact with one another.

# **UAPP Architecture Design**

**Apache Architecture**



**Dockerized Architecture**

****

# **U-APP Installation**

# **Repository**

* Both mobile and website codebase are in one repository just in different branches so it is important that these main branches **MUST NOT** **MERGE**..

Repo: <https://gitlab.com/danielaplewe/u-app_codebase_2023_nov_onwards_current.git>

# **Main Development Branches**

# **main**

* + Root branch but empty

# **uapp-mobile**

* + Main branch of mobile development

# **uapp-web**

* + Main branch of website development

# **UAPP-WEB-EDU**

* + Main branch for website development for Edu variant

# **Main Production Branches**

# **uapp-web-production**

* + Main branch for website production codebase
  + Can be updated by merging the main web branch to this.

# **Website Main Plugin Version**

* Laravel ^9.11
* Php ^8.1
* Mysql ^5.7

# **Mobile Main Plugin Version**

* React-Native ^0.75.5
* Expo ^49.0.13

# **Local Development**

# **Web Setup \* Local**

*Ensure that you are on the master branch.*

* Open the code base in Visual Studio Code.
* Create a duplicate of the **.env.example** file, and subsequently eliminate the **.example** extension from the duplicated file.
* Ensure the completion of the Databases and Admin fields and other required fields within the **.env** file.(**See next page for the .env example**)
* Create a duplicate of the **docker-compose.yml.example** file, and subsequently eliminate the **.example** extension from the duplicated file.
* Upon modifying the **.env** file and creating the **yml** file, execute the command ‘**docker compose up’** in the terminal, ensure that your directory is in the codebase.
* Once the containers are successfully created and running, proceed with the command ‘**docker exec -it mysql mysql -uroot -proot’.**
  + Run ‘**show databases’** to see the database lists
  + If there is no database named “**uapp**” create one by running **‘create database uapp’**
  + If there is no database named “**uapp\_activity**”, create one by running **‘create database uapp\_activity’.**
  + Exit the MySQL container.
* Post-exit, execute the command ‘**docker exec -it uapp-php-fpm bash’**.
* Run the command ‘**composer install’** following the installation of dependencies.
* Generate a key using the command ‘**php artisan key:generate’**.
* Execute php artisan config:clear and ‘**php artisan migrate’** along with ‘**php artisan db:seed’.**
* Adjust the permissions of the storage folder by executing the command **‘chmod 777 -R storage’**.
* You should now be able to access the U-APP website utilizing the credentials provided in the **Admin fields** of the **.env** file.

**Note:**

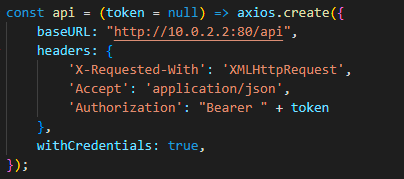
* In case of encountering an error in running php artisan commands, attempt resolving it by executing ‘**docker exec -it uapp-php-fpm update-alternatives --set php /usr/bin/php8.1’**.
* Ensure to run ‘**docker exec -it uapp-php-fpm php artisan config:clear’** whenever modifications are made to the **.env** file or the configuration folder.

# **Website .env Example**

* APP\_NAME=Laravel
* APP\_ENV=local
* APP\_KEY=
* APP\_DEBUG=true
* APP\_URL=http://localhost
* APP\_DOMAIN\_NAME=localhost
* LOG\_CHANNEL=stack
* LOG\_DEPRECATIONS\_CHANNEL=null
* LOG\_LEVEL=debug
* DB\_CONNECTION=mysql
* DB\_HOST=mysql
* DB\_PORT=3306
* DB\_DATABASE=uapp
* DB\_USERNAME=root
* DB\_PASSWORD=root
* DB\_CONNECTION2=mysql
* DB\_HOST2=mysql
* DB\_PORT2=3306
* DB\_DATABASE2=uapp\_activity
* DB\_USERNAME2=root
* DB\_PASSWORD2=root
* BROADCAST\_DRIVER=log
* CACHE\_DRIVER=file
* FILESYSTEM\_DISK=local
* QUEUE\_CONNECTION=sync
* SESSION\_DRIVER=file
* SESSION\_LIFETIME=120
* MEMCACHED\_HOST=127.0.0.1
* REDIS\_HOST=127.0.0.1
* REDIS\_PASSWORD=null
* REDIS\_PORT=6379
* MAIL\_MAILER=smtp
* MAIL\_HOST=mailhog
* MAIL\_PORT=1025
* MAIL\_USERNAME=null
* MAIL\_PASSWORD=null
* MAIL\_ENCRYPTION=null
* MAIL\_FROM\_ADDRESS="hello@example.com"
* MAIL\_FROM\_NAME="${APP\_NAME}"
* ADMIN\_FIRST\_NAME=admin
* ADMIN\_LAST\_NAME=uapp
* ADMIN\_USERNAME=admin\_uapp
* ADMIN\_EMAIL=admin\_uapp@gmail.com
* ADMIN\_PASSWORD=password12345
* AWS\_ACCESS\_KEY\_ID=
* AWS\_SECRET\_ACCESS\_KEY=
* AWS\_DEFAULT\_REGION=us-east-1
* AWS\_BUCKET=
* AWS\_USE\_PATH\_STYLE\_ENDPOINT=false
* GOOGLE\_CLIENT\_ID=
* GOOGLE\_CLIENT\_SECRET=
* PUSHER\_APP\_ID=
* PUSHER\_APP\_KEY=
* PUSHER\_APP\_SECRET=
* PUSHER\_APP\_CLUSTER=mt1
* MIX\_PUSHER\_APP\_KEY="${PUSHER\_APP\_KEY}"
* MIX\_PUSHER\_APP\_CLUSTER="${PUSHER\_APP\_CLUSTER}"

# **Mobile App Setup \* Local**

*Make sure you are on the `master` branch.*

* **Basic Setup**
  + Open code base in Visual Studio Code.
  + Open the `**api.ts**` file under services directory.
    - 
    - *Make sure you are connected using your local IP address with port 80.*
  + Create a duplicate of the **docker-compose.yml.example** file, and subsequently eliminate the **.example** extension from the duplicated file.
  + Ensure in the **docker-compose.yml** file the **REACT\_NATIVE\_PACKAGER\_HOSTNAME** is the current ***local IP address***  of your computer
    - **
  + Run command `**docker compose up**` on the codebase terminal.
  + After the container is created and running proceed with the command `**docker exec -it uapp-expo npm install**`.
  + After that run the app by running the command ‘**docker exec -it uapp-expo expo start’**
    - ****
    - **it will show the qr code needed to connect to external devices for testing**
* **Connecting to External testing device (Android)**
  + Download **Expo Go** in your Android phone
  + Open the app the click **‘Scan QR code**’ in the selection
  + Scan the qr shown in the terminal to run the app.

# **Production Development**

# **Domains**

* Main Domain for Backend Website
* Api.u-app.com.sg (Currently using Apache)
* Elastic IP: 18.142.123.240

# **Ec2 Configuration**

* Platform: **Ubuntu**
* Inbounds Port
  + 80 - \*
  + 443 - \*
  + 22 - preferred to put your own ip for security purposes
* Outbounds Port
  + All Traffic - \*

# **Database**

* Currently in Ec2
* Database Type: Mysql
* Main Database Name : **uapp**
* Secondary Database Name: **uapp\_activity**

# **File Storage**

* Currently using Ec2 volume.
* Located inside the storage folder of the codebase.

# **Web Setup \* Production Deployment (Apache)**

**1. AWS setup:**

- For Apache2 , HTTPS will be hosted at port 443 and HTTP will be hosted at port 80, so we will need to allow inbound connection to those ports => Setting VPC security group and allow inbound connections.

**2. Inside the EC2 instance:**

- Configure the firewall and allow connection to ports 443 and 80.

- Using Git (preferred) or SSH and downloading the code to a designated location.

- Install and create a production build of the project

- Configure Apache2 (Nginx) and install all necessary extensions.

- Configure Apache2 (Nginx) permission in the project folder.

- Configure the SSL certificate for the domain to enable HTTPS (preferred using Certbot)

- Configure Apache2 (Nginx) config for HTTP and HTTPS and point it to the project folder.

- Restart Apache2 (Nginx).

# **Web Setup \* Production Deployment (Dockerized)**

1. Make sure your Ec2 instance is ready and it has the right configurations
   1. Ubuntu
   2. Security Inbound Ports
      1. 80 - \*
      2. 443 - \*
      3. 22 - preferred to put your own ip for security purposes
   3. Security Outbound Ports
      1. All Traffic - \*
2. Ensure you have the **UAPP-WEBSERVER-KEY-PAIR.pem**
3. Locate the pem file in your computer and ssh to the server by doing the command below.
   1. ssh -i /path/to/pem/file ubuntu@[ec2 ip]
4. Once inside make sure you have installed the following services, refer to the installation guidelines found on the internet.
   1. Docker
   2. Screen
5. Setup the codebase
   1. In using git make sure you have an account that have the permission to pull from the remote repo.
   2. git clone <https://gitlab.com/danielaplewe/u-app_codebase_2023_nov_onwards_current.git>
   3. cd u-app\_codebase\_2023\_nov\_onwards\_current\_git
   4. git fetch origin uapp-web-production
   5. git checkout uapp-web-production
6. Create a duplicate of the **.env.production.example** file, and subsequently eliminate the **.production.example** extension from the duplicated file.
   1. Ex. **cp .env.production.example .env**
7. Create a duplicate of the **docker-compose.yml.production.example** file, and subsequently eliminate the **.production.example** extension from the duplicated file.
   1. Ex. **cp docker-compose .yml.production .docker-compose.yml**
8. Ensure the .env file has the right values such as the keys and urls. Upon modifying the **.env** file and creating the **yml** file, execute the command ‘**sudo docker compose up’** in the terminal, ensure that your directory is in the codebase.
9. Once the containers are successfully created and running, proceed with the command ‘**sudo docker exec -it mysql mysql -uroot -proot’.**
   1. Run ‘**show databases’** to see the database lists
   2. If there is no database named “**uapp**” create one by running **‘create database uapp’**
   3. If there is no database named “**uapp\_activity**”, create one by running **‘create database uapp\_activity’.**
   4. Exit the MySQL container.
10. Post-exit, execute the command ‘**sudo docker exec -it uapp-php-fpm bash’**.
11. Run the command ‘**composer install’** following the installation of dependencies.
12. Generate a key using the command ‘**php artisan key:generate’**.
13. Execute php artisan config:clear and ‘**php artisan migrate’** along with ‘**php artisan db:seed’.**
14. Adjust the permissions of the storage folder by executing the command **‘chmod 777 -R storage’**.
15. You should now be able to access the U-APP website utilizing the credentials provided in the **Admin fields** of the **.env** file.
16. Test the production site(best if you change the .env **APP\_ENV** to local and **APP\_DEBUG** to true
17. After testing change the.env **APP\_ENV** to production and **APP\_DEBUG** to false

**Note:**

* In case of encountering an error in running php artisan commands and composer commands, attempt resolving it by executing ‘**sudo docker exec -it uapp-php-fpm update-alternatives --set php /usr/bin/php8.1’**.
* Ensure to run ‘**sudo docker exec -it uapp-php-fpm php artisan config:clear’** whenever modifications are made to the **.env** file or the configuration folder.

# **U-APP Maintenance Guide**

# **Production Web Server Update Guide(dockerized)**

* Ensure you have the **UAPP-WEBSERVER-KEY-PAIR.pem**
* Locate the pem file in your computer and ssh to the server by doing the command below.
  + ssh -i /path/to/pem/file ubuntu@[ec2 ip]
* Enter the codebase folder
* Pull the production branch from remote
  + git pull origin uapp-web-production
* Do the necessary in the following commands in succession
  + **sudo docker exec -it uapp-php-fpm update-alternatives --set php /usr/bin/php8.1’**
  + **sudo docker exec -it uapp-php-fpm composer install** (Needed if there are changes in the packages used)
  + **sudo docker exec -it uapp-php-fpm php artisan migrate** (Needed if there are changes in the migration)
  + **sudo docker exec -it uapp-php-fpm php artisan db:seed --class=YourSeederClass** (Do this if there are new seeder class)
  + **sudo docker exec -it uapp-php-fpm php artisan config:clear**
* Verify changes in the production website

# **Notes**

ADMIN\_USERNAME=admin\_uapp

ADMIN\_EMAIL=admin\_uapp@gmail.com

ADMIN\_PASSWORD=4Rh1yqSa