**Setting up the assignment project Competition Management System**

There are 5 important files. The **Log.txt** is a file used by the author to *recall* the file content version.

The **backend.zip**

It is a NodeJS backend project which supports the client side REST API calls.

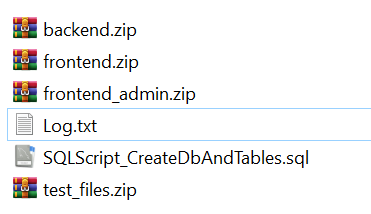
The **frontend.zip**

It is a client-side ReactJS project which supports the client-side normal user functionalities.

The **frontend\_admin.zip**

It is a client-side ReactJS project which supports the client-side administrator functionalities.

The **SQLScript\_CreateDbAndTables.sql** is a script file. The MySQL requires this file's SQL commands to create tables and test records.

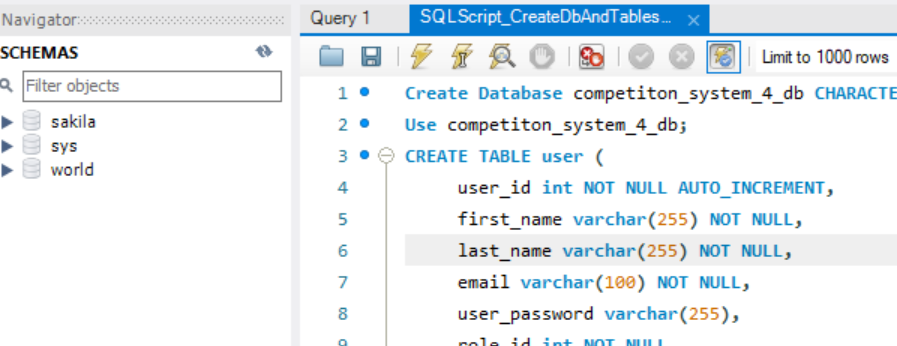


**Setup the database**

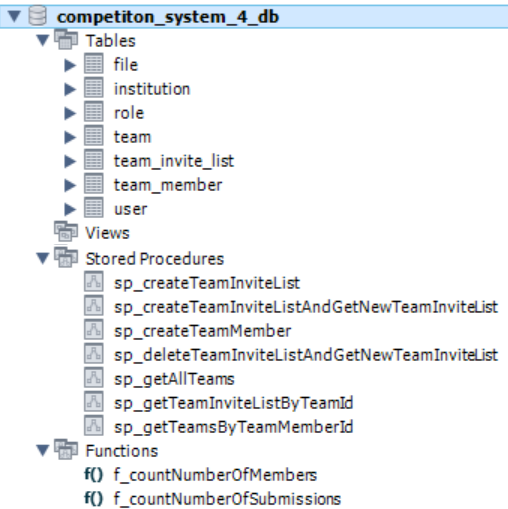
Start MySQL Workbench.



Open the SQL script file. Execute the SQL script file.



Observe the schema interface at the MySQL Workbench. The SQL script file has commands which define the tables, stored procedures and functions.



**Note:**

Observe the SQL command below. The **created\_at** field is defined by using TIMESTAMP data type. Research why this data type is needed and how this data type will impact your development work.

CREATE TABLE user (

user\_id int NOT NULL AUTO\_INCREMENT,

first\_name varchar(255) NOT NULL,

last\_name varchar(255) NOT NULL,

email varchar(100) NOT NULL,

user\_password varchar(255),

role\_id int NOT NULL,

institution\_id int NOT NULL,

status varchar(50) NOT NULL,

created\_at TIMESTAMP DEFAULT now(),

PRIMARY KEY (user\_Id),

UNIQUE (email)

)AUTO\_INCREMENT=100;

**Setup the backend project by modifying the .ENV file content**

Extract all the three project zip files.

This section focuses on **backend.zip** file. Extract this zip file.

Use Visual Studio Code to focus on the **backend** directory which is extracted from the **backend.zip**.

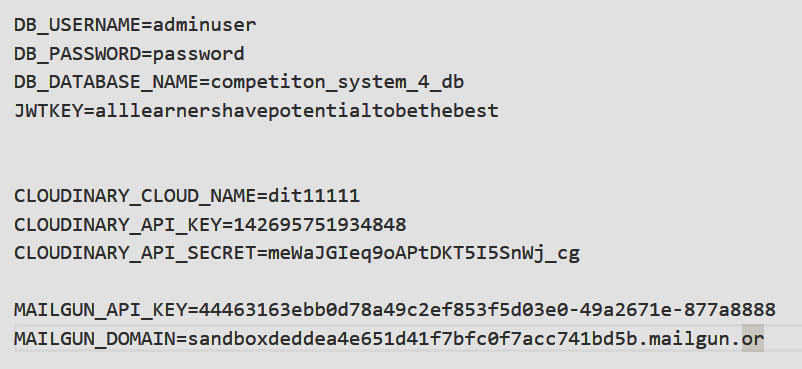
Open the **.ENV** file.



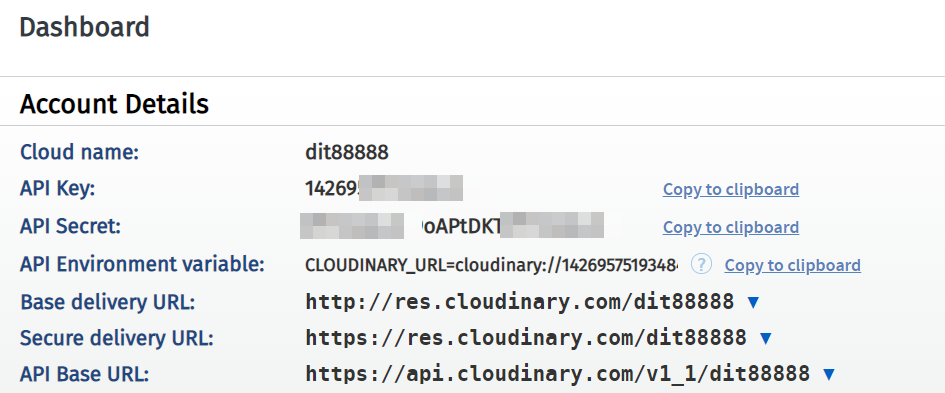
**Setup Cloudinary to support file management functionalities**

The 1st four lines of key-value information does not require any modifications.

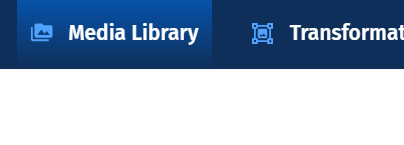
Create a Cloudinary account at **www.cloudinary.com**.



Logon to Cloudinary. At the Dashboard interface, apply your cloud name, API key and API secret information in the **.ENV** file. The backend project code reads this key-value pair information to manage files inside the Cloudinary repository.

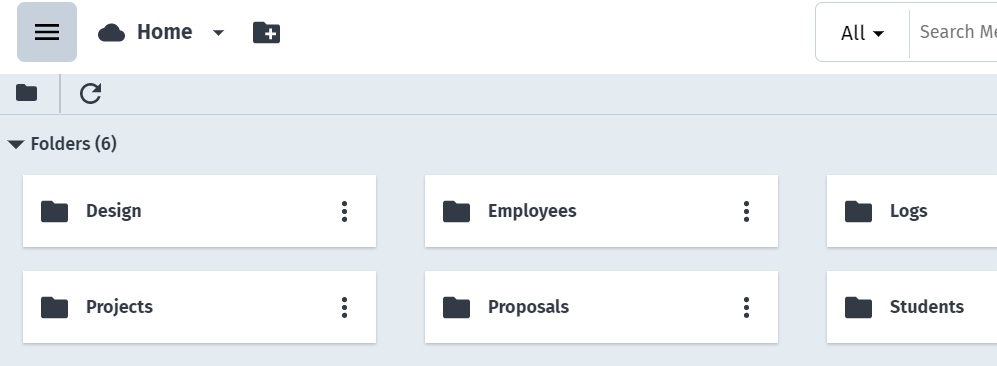


Visit the Media Library interface.

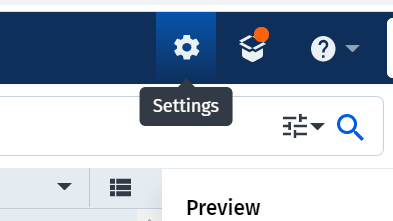


Create a directory, **Proposals**.

The code inside the backend project targets on this directory to save and delete files.

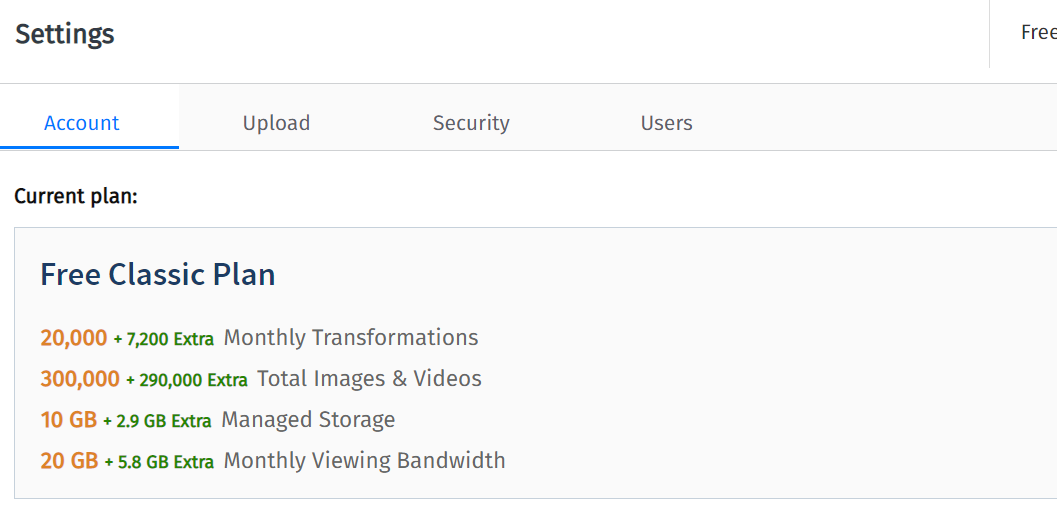


At the Media Library interface's top right hand corner, click **Settings**.



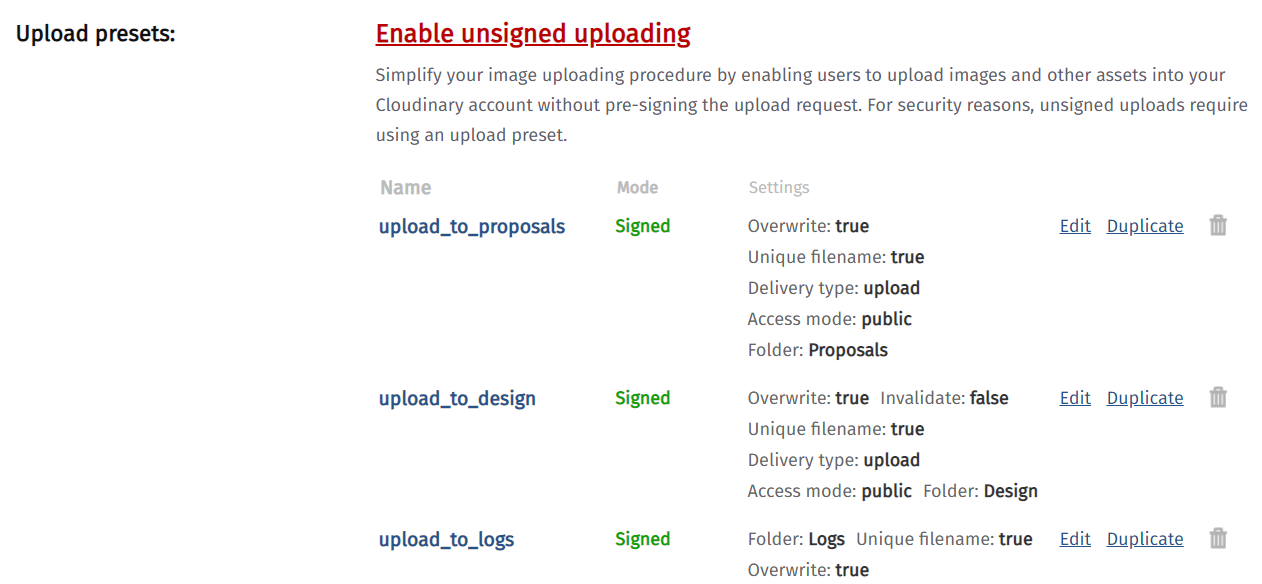
At the Settings interface, you need to configure the Cloudinary on how your backend project can upload and manage files inside the **Proposals** directory which was created in the earlier steps.

**Click** the **Upload**.

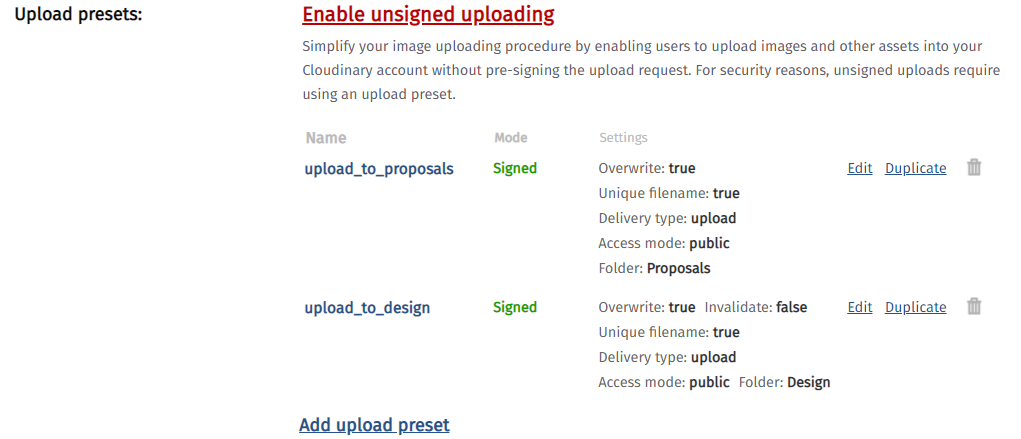


Scroll down to find the section, **Upload presets**. You will create one "upload preset". The name of the "upload preset" will be **upload\_to\_proposals**.

Click the **Add upload preset**.



Creating a **preset** *simplifies* the code which does the file management operations.



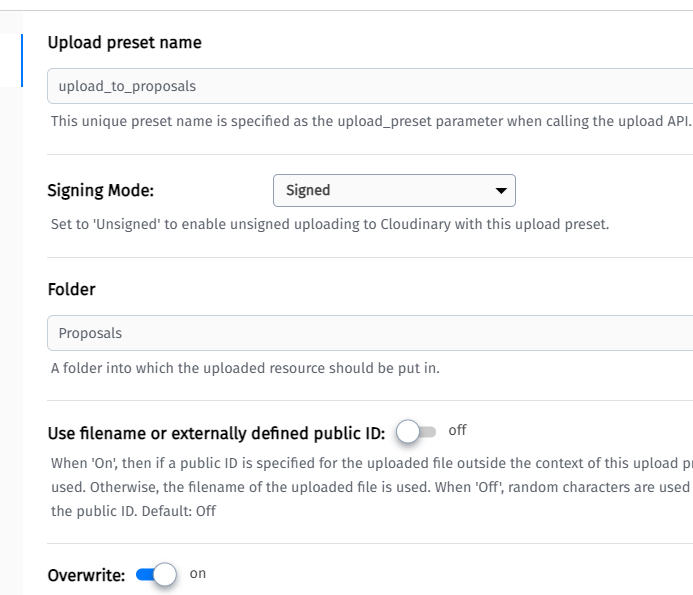
Click the **Add upload preset** to begin setting up a new upload preset.

Only **two inputs** are required:

① Provide the upload preset name, **upload\_to\_proposals**.

② Set the upload preset to focus on the folder, **Proposals**.

You use the rest of the settings as **default**.



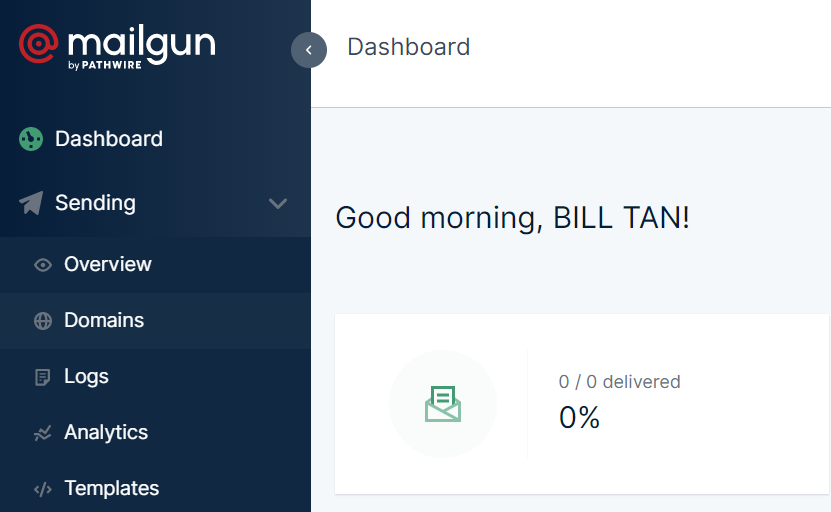
**Setting up MailGun to support system's email functionalities**

The backend project has a file, **\controllers\userController.js**. The file has code which uses MailGun to send emails by calling the MailGun API.

**Video reference** on setting up MailGun: [**https://youtu.be/Xnf\_Hw\_-UiU**](https://youtu.be/Xnf_Hw_-UiU)

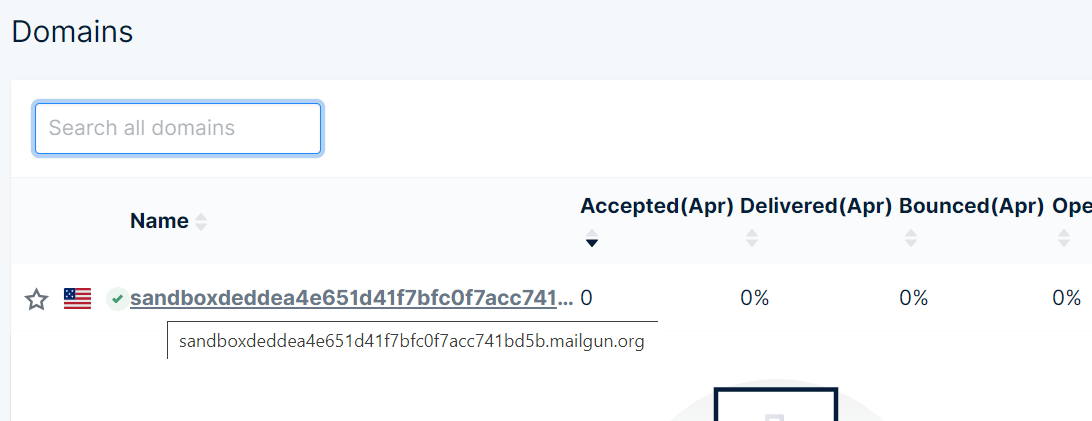
Create a user account in MailGun.

Within the menu, expand the **Sending**. Then, choose **Domains**.

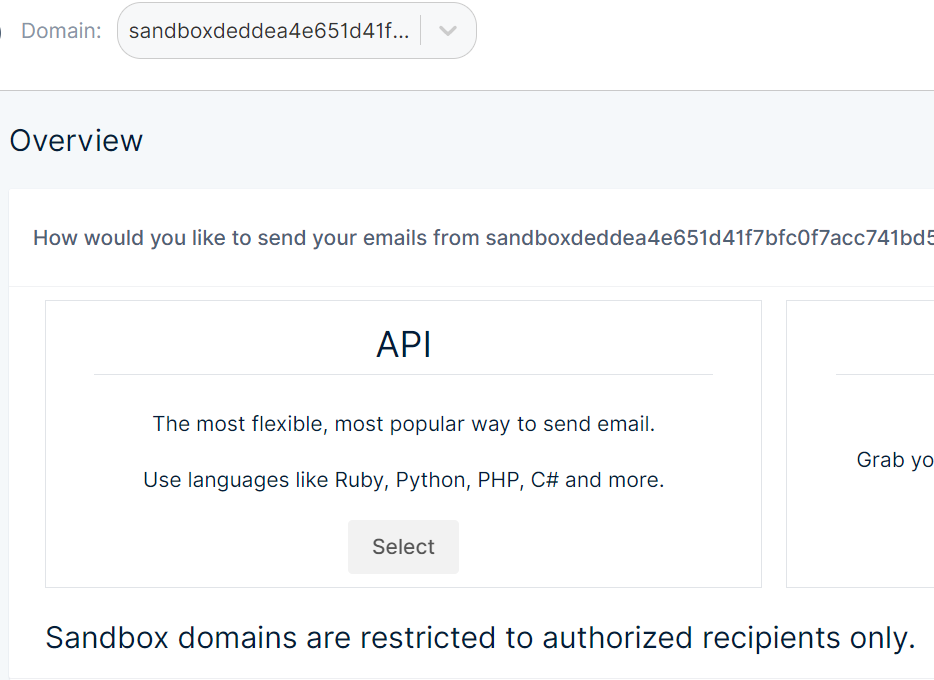


You should see a sandbox domain. Refer to the article at [**https://blog.mailtrap.io/mailgun-sandbox-tutorial/**](https://blog.mailtrap.io/mailgun-sandbox-tutorial/%20) to appreciate how sandbox domain works. The article's content emphasizes that you need to create **verified recipients** to experiment email sending process.

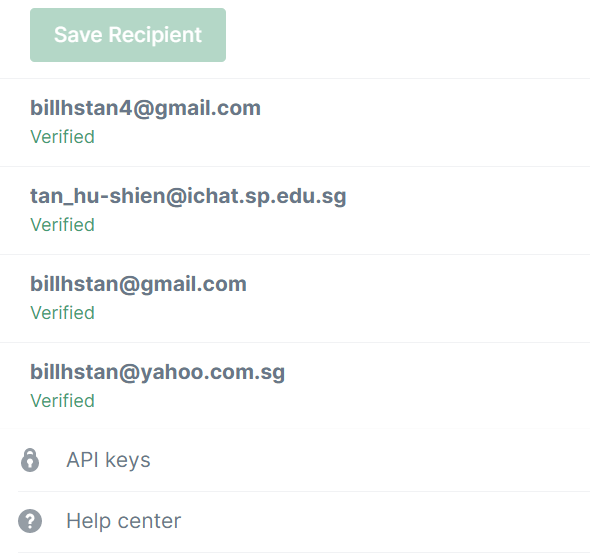
To begin creating verified email accounts, **click** the domain link to access the domain management interface. Note that the domain name is required for your project's **.ENV** file.



At the domain management interface, **scroll down** to find the authorized recipient section.



Use your own **ichat.sp.edu.sg** type email accounts to create **verified** recipients. Note that, type accounts are preferred.



Click this **API keys** to obtain your API key information for your backend project's **.ENV** file.

**Finally, test the competition management system project.**