

Thank-A-Teacher

Installation Guide

Version 1.0

12/2/24

Table of Contents:

Prerequisites:	3
1. Install Node.js and npm	3
2. Clone the Repository	3
3. Set up MongoDB	3
4. Set up EmailJS	3
Installation:	5
1. Install Dependencies	5
2. Set up Emailjs code	5
Running the Application:	7
a. Start the database	7
b. Start the React application	7
Troubleshooting:.....	8
Missing Dependencies	8
TA Search not Populating	8
TA Inbox not populated with sent cards	8
Email not being sent to the TA account.....	8

Prerequisites:

1. Install Node.js and npm

a. Download and install Node.js: <https://nodejs.org/en>

b. Confirm Installation:

```
node -version
```

```
npm -version
```

2. Clone the Repository

c. Run the lines below in your preferred location:

```
git clone https://github.com/CS-3311-Group-4115/Thank-A-Teacher.git
```

```
cd Thank-A-Teacher
```

3. Set up MongoDB

a) Create a MongoDB account at this link: <https://www.mongodb.com/>

b) Create a Cluster

c) Select “Connect” and then “Drivers”

a. From this menu, you can find the “Connection String”

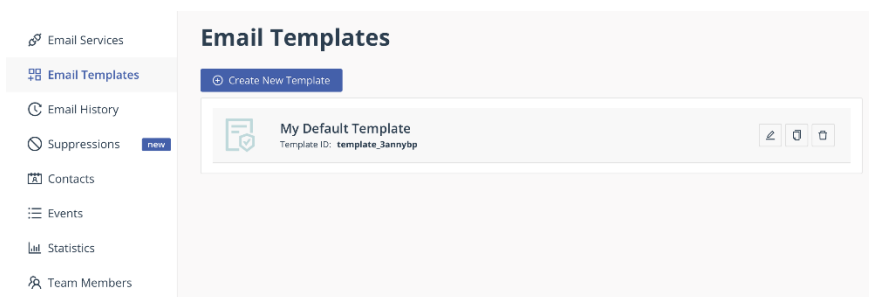
b. Place this “connection string” in backend/thank-a-teacher/server.js in the uri variable

4. Set up EmailJS

d. Create an account at <https://www.emailjs.com/>

e. Select ‘**Email Templates**’ on the left Navigation

f. Select ‘**Create New Template**’



- g. On the template content page, edit all required portions (subject, content, to email, from email). Ensure that all curly bracketed information keeps the naming as seen in the image below.

The screenshot shows the 'My Default Template' editor. At the top, there are tabs for 'Content', 'Auto-Reply', 'Attachments', 'Contacts', and 'Settings'. The 'Content' tab is active. The main editing area is divided into two columns. The left column contains the 'Subject' and 'Content' fields. The 'Subject' field has the text 'New message from {{from_name}}'. The 'Content' field has a preview of the email body, which includes a greeting 'Hello {{to_name}}.', a notification 'You got a new message from {{from_name}}:', a placeholder for the message content '[[message]]', and a sign-off 'Best wishes, thankateacher'. There are also 'Desktop' and 'Mobile' view toggles and an 'Edit Content' link. The right column contains the 'To Email', 'From Name', 'From Email', 'Reply To', 'Bcc', and 'Cc' fields. The 'To Email' field has the placeholder '{{to_email}}'. The 'From Name' field has the text 'jessie'. The 'From Email' field has a checked checkbox for 'Use Default Email Address' and a placeholder. The 'Reply To' field has the placeholder '{{reply_to}}'. The 'Bcc' and 'Cc' fields are empty.

My Default Template [Playground](#) [Test It](#) [Save](#)

Content Auto-Reply Attachments Contacts Settings

Subject *
New message from {{from_name}}

Content *
Desktop Mobile Edit Content

Hello {{to_name}},
You got a new message from {{from_name}}:
[[message]]
Best wishes,
thankateacher

To Email *
{{to_email}}

From Name
jessie

From Email *
☒ Use Default Email Address

Reply To
{{reply_to}}

Bcc

Cc

- h. Save and Exit

Installation:

1. Install Dependencies

- Run the following command in the project root to install all required packages.

```
npm install
```

- For any dependency errors/library references refer to Troubleshooting

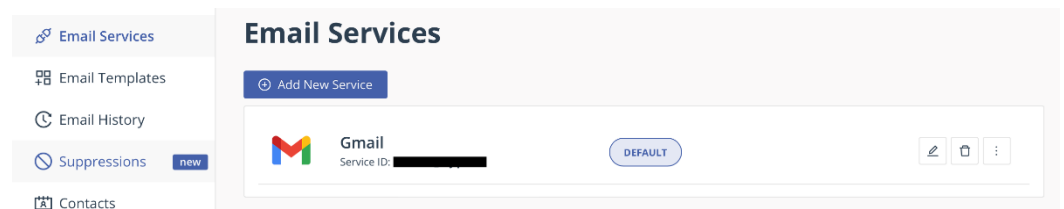
2. Set up Emailjs code

- In your local repository, navigate to basepage.js.
- At line 89 you will find the code associated with sending emails to teaching assistants.

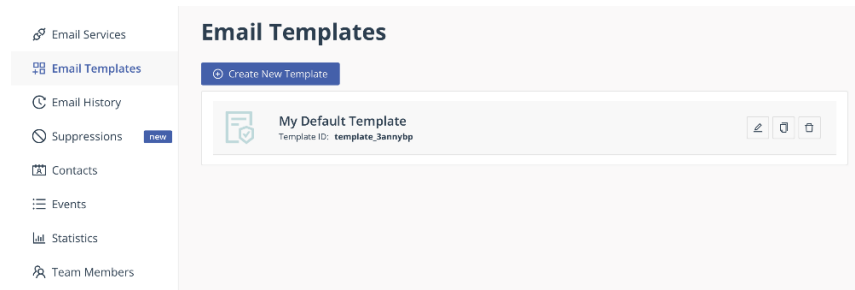
```
89     if (confirmSend) {
90       const message = "You have been sent a card! http://localhost:3000/login"; // Assuming 'text' contains the card message
91       const cardImage = cards[selectedCard - 1]; // Get the selected card image
92
93       const templateParams = {
94         to_email: selectedTAEmail, // Recipient email
95         from_name: 'thankateacher', // Sender name (could be dynamic)
96         message: message,
97       };
98
99       emailjs.send('', '', templateParams, '')
100         .then((response) => {
101           console.log('Email successfully sent!', response.status, response.text);
102           alert('Email sent successfully!');
103         })
104         .catch((error) => {
105           console.error('Failed to send email:', error);
106           alert('Failed to send email.');
```

- Edit the four parameters on line 99 with the information associated with your created template and account.

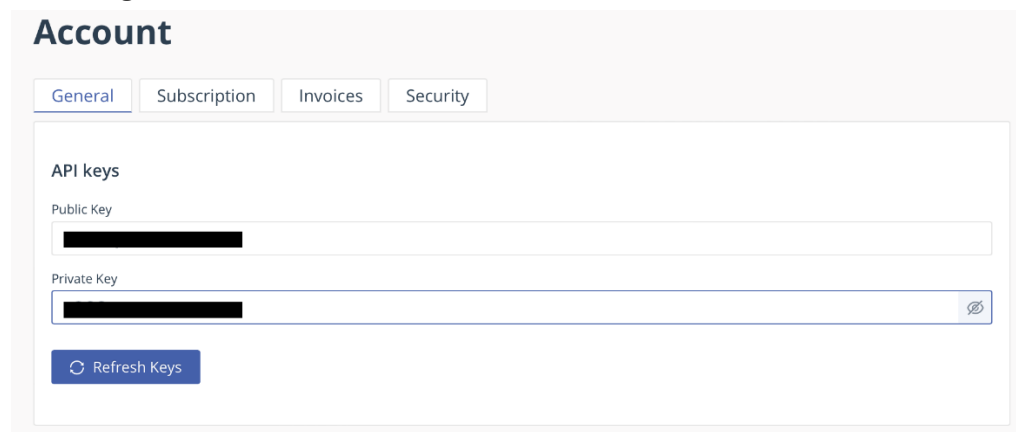
- The first parameter should be the email service ID associated with the account. This can be found at the **'Email Services'** Tab on the left navigation.



- The second parameter should be the Template ID. This can be found after selecting **'Email Templates'** Tab.



- iii. The third parameter should remain the same: **'templateParams'**
- iv. The fourth parameter is the **public** key associated with your EmailUS account. This can be found by navigating to the **'Account'** tab on the left navigation.



- v. Save your changes to basepage.js

3. Setup MongoDB format

- a. The MongoDB is setup with two collections and can be found when looking at the collections of Cluster0.
- b. Upon logging in to MongoDB you should see the following Cluster:

▼ **thank-a-teacher**

CARD

TA

- i. The thank-a-teacher is the database with CARD and TA are collections
 - 1. **CARD** stores the jpeg strings associated with the sent cards
 - 2. **TA** stores information about teaching assistants including names and email addresses.
- c. User accounts are auto generated upon registration and are stored in the **'system.users'** collection which is located in the admin database
 - i. This is automatically set up by MongoDB

Running the Application:

a. Start the database

- a. cd into your repository location
- b. cd into 'backend/thank-a-teacher'
- c. Run the following command
`npm start`

b. Start the React application

- a. cd into your repository
- b. cd into 'ta-thanks'
- c. Run the following command
`npm start`
- d. Open your web browser and navigate to <http://localhost:3000>

Troubleshooting:

Missing Dependencies

- a. Run `npm start` again to ensure that all dependencies are installed.
- b. If a specific dependency is causing errors, download it directly by running:
`npm install package-name`
 - i. Relevant dependencies:
 - i. `npm install html2canvas`
 1. <https://html2canvas.hertzen.com/>
 - ii. `npm install axios`
 1. <https://axios-http.com/docs/intro>
 - iii. `npm install emailjs`
 1. <https://www.emailjs.com/docs/sdk/installation/>
 - iv. `npm install react-draggable`
 1. <https://www.npmjs.com/package/react-draggable>

TA Search not Populating

- a. Ensure that the database is running alongside the React application.
- b. If the React application was started before the database – refresh the page.

TA Inbox not populated with sent cards

- a. Ensure that when the card is sent it is received in the database terminal.
 - a. The ID string associated with the card should be shown in the terminal.
- b. Make sure that the account being logged into is marked 'isTA' in the MongoDB database.
 - a. After editing the database, restart the application and login to the TA account.

Email not being sent to the TA account

- a. Ensure that you are logged into an account before sending any cards through the application.
 - a. Also ensure that a TA has been selected in the search menu.
- b. Check the associated parameters with the EmailUS call and ensure that they use the correct template ID, emailID, and public key.

- a. Also ensure that the templateParameters are filled out and are managed correctly.
- c. Test the template in your EmailUS account by navigating to the '**Create Template**' page and selecting edit.
 - a. You can send a test email here by selecting the '**Test It**' button.
- d. Check the logs on your EmailUS account by selecting '**Email History**' on the left navigation.