**Moon and Cake Team Roles**:

Max Yeh: Product Owner

Yao-Wen Chang: Scrum Master (rotated)

Jinson Wu

Yi-Chia Wu

Ya-Ru Yang

**Customer Meeting Date/Time/Place**:

weekly meeting: 10/28, 11/04 via Zoom (9:00-10:00 am every Friday via Zoom)

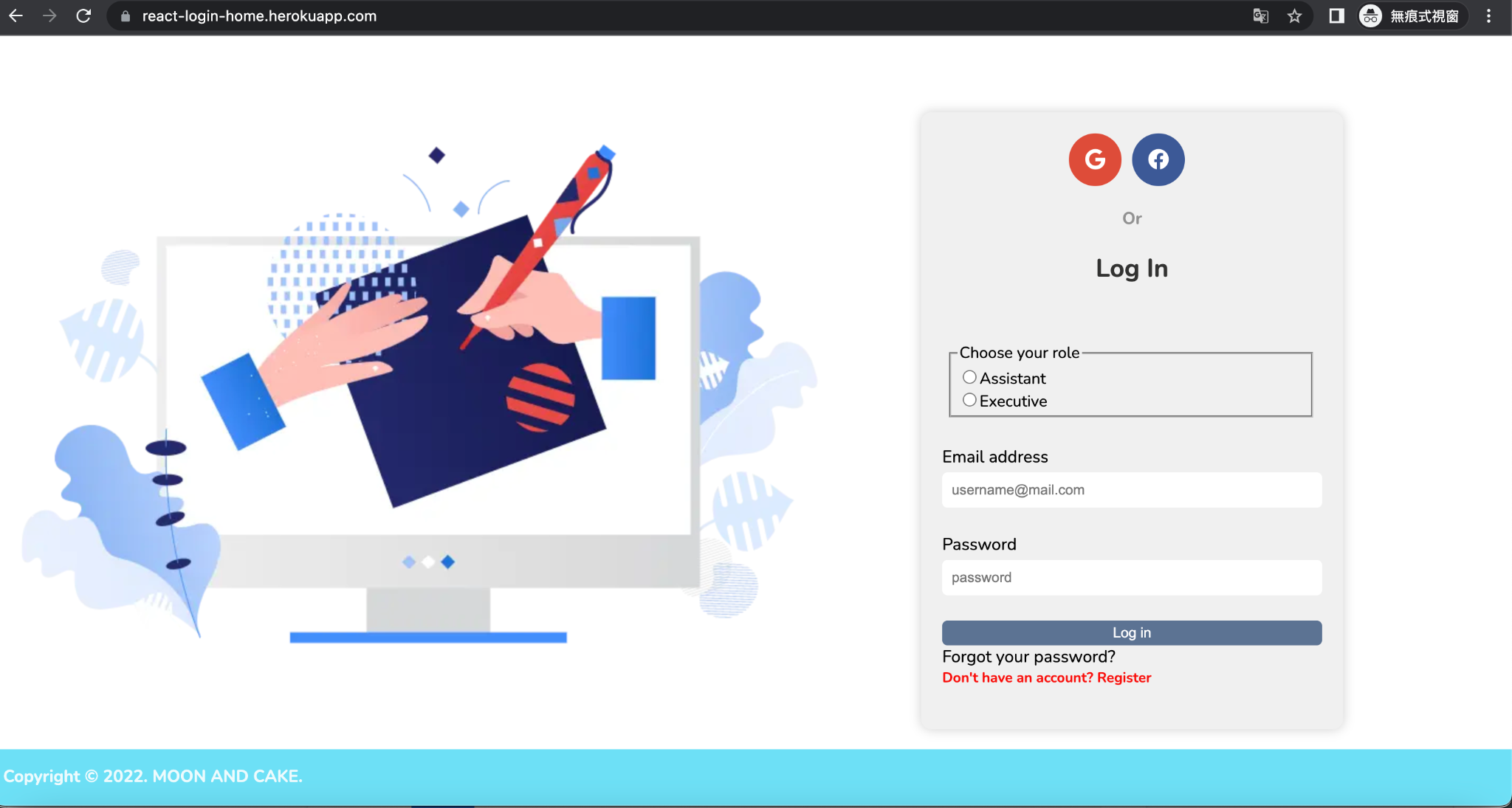
**Feedback from the Client (mainly from the meeting on 11/04):**

* The primary tasks in the next iteration are executive’s interface design, draft audio files queueing implementation and email authorities assignment.
* Completed assistant’s interface design and database integration
* Homepage, login/signup page, and the two-type login worked well and passed tests perfectly

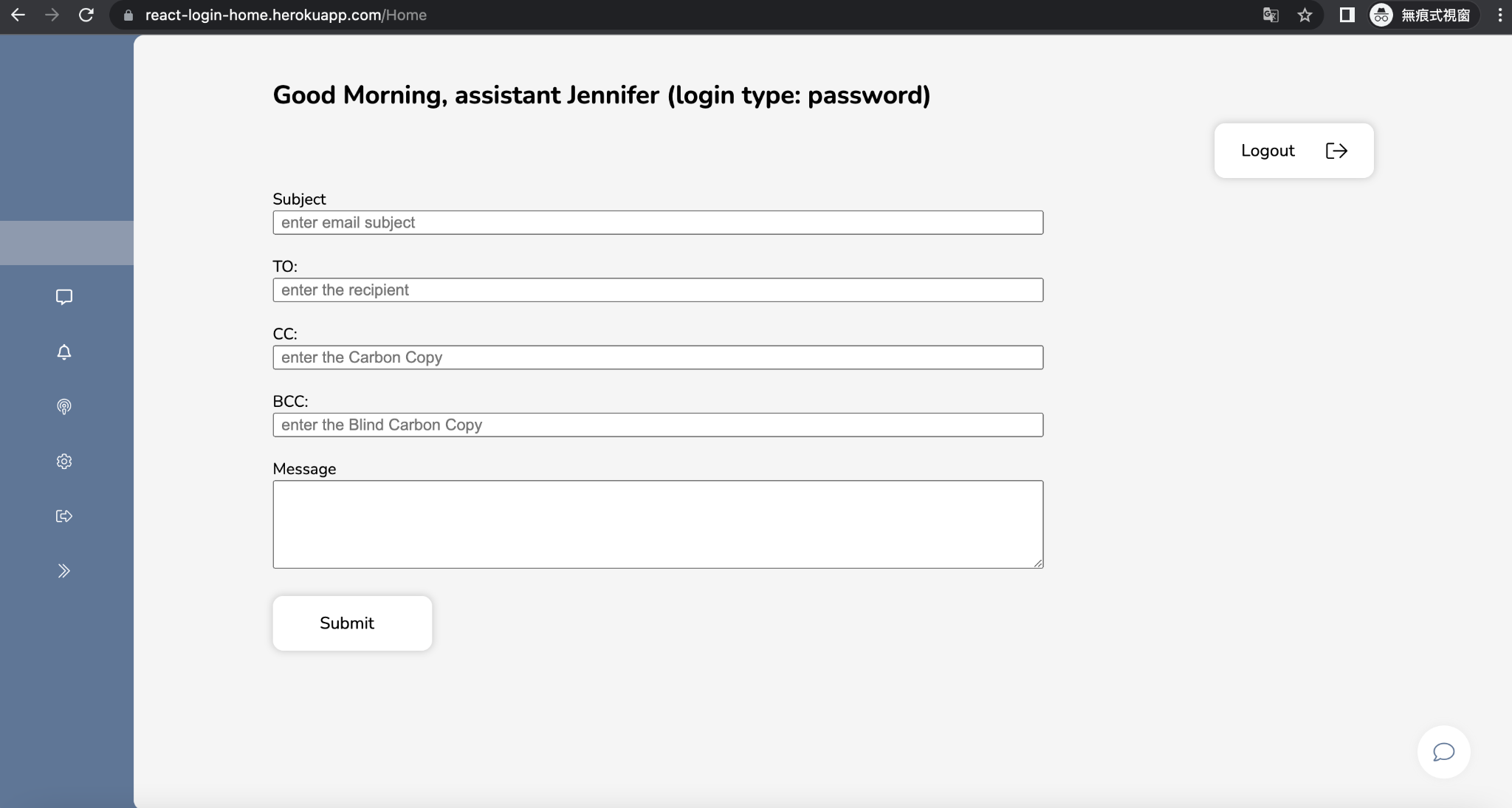
**Current Working Stories**:



* login via third-party authentication and website registration, assistant’s homepage design, front-end deployment on Heroku, and back-end deployment on Heroku.
  + New user stories (Add/Delete draft function, draft audio files queueing, and email authorities assignment) were added which could better describe the requirement of the client.
  + Latest details of the two-type login separation and assistant’s homepage design were proposed in the meeting last week. We are stepping on these added features to meet the requirements and deploying them on the platform.
  + This iteration added a role selection checkbox on the login interface. Also System by React and Firebase-Django back-end coordination employed on Heroku, which could identify different users while logging into the website. The role type shows on the left-top corner after the greeting message.
  + Primary login page and the assistant’s home page design are shown with the functions demonstrated below. Internal back-end integration was completed and currently under the test.
  + Database linked with the user profile is under construction. We now emphasize the combination of backend and add-on profiles within Django framework

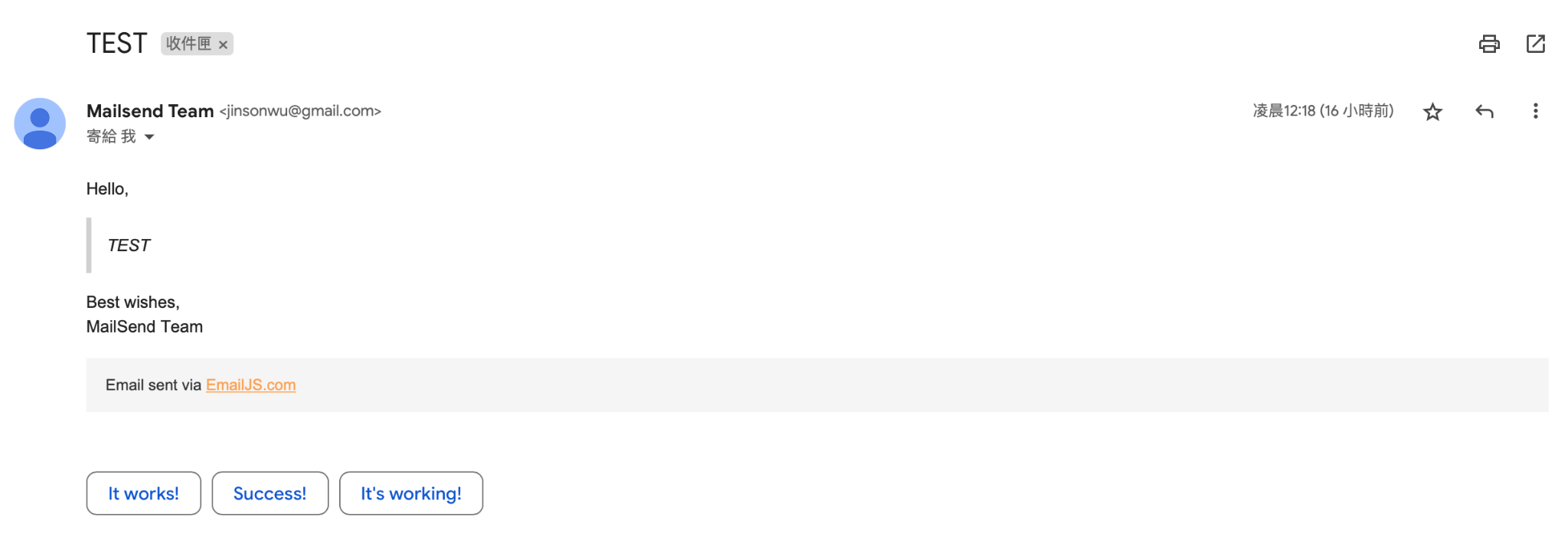


Login/registration page



Homepage with form sending emails

* Backend Integration with Django and Firebase:
  + Horizontally transferred the current serverless backend from Firebase to Django.
    - Firebase was directly performed by Google API. We created credentials from Google developer platform to utilize the authentication and Gmail function on our own.
    - Included credentials and API Key in the frontend implemented by React.JS to certify semantically functional equivalent.
  + Concatenated the original frontend executed by React.JS with the transferred backend
    - Designed the main visualization and padded all the requirements the client desired.
    - Fetched the user information for subsequent operations through form or board .
  + Database by Firebase
    - Although we transferred our backend to Django, realtime database provided by Firebase was convenient for internal communication and further linkage.
    - Firestore database could store user profile information through third-party authentication and website registration. We could acquire the data we demanded from that.
* MailSending Feature:
  + Sent emails via EmailJS
    - Registered the service of EmailJS to accomplish mail sending functions and employed it in our application.
    - Drafts could firstly be composed by assistants and sent to executives. They could customize content, subject, address, and etc., with the form we provided.



Brief look of the mail sending by us (MailSend Team)

* + Created form to obtain necessary information
    - Redesigned the homepage and a new tab containing the form used for sending emails.
* Test case:
  + Cucumber and RSpec test cases:
    - This will be created after accomplishing the dashboard.
  + BDD: Test an application's behavior from the end user
    - There will be two ways for users to log in, one is logged by a google account and the other is by an email-signup account.
    - User can choose his login role type on the checkbox
    - Different selected roles will lead to different homepage after login (pop-up message then redirect).
    - Mail sending form would vary depending on the logged in roles
  + TDD: testing smaller pieces of functionality
    - type 1: enter username, password and select login role on the checkbox
    - type 1: frontend passes the information and interacts with database via backend service
    - type 1: store user information in database
    - type 2: able to access google account (into account selection page)
    - type 2: use google account to login and fetch the basic information from the platform
    - different role types and username give different result (text show on the left-top corner after the greeting message)
    - both: information stored in the database
    - both: mail sending test (whether the mails correctly deliver/receive)
  + Evaluation (in the next several iterations)
    - Executive’s Dashboard construction (main visualization optimization)
    - Draft functions and email authorities assignment should be completed in the next few iterations
    - Database reliability test after fully constructed
    - Notifying window including audio files works or not
* UML class diagram

