# WeMeet

COMS E6998 Cloud Computing and Big Data - Project

Wannuo Sun (ws2591), Jiaqing Chen (jc5657), Lukas Wang (bw2712), Yang Li (yl4850)

May, 2022

**Abstract**

In the wake of the epidemic, online meetings have played a crucial role in all aspects of education, economy, and society. However, existing meeting applications only care about the meeting function itself, and there is no proper solution for meeting schedule and member time conflicts, so we propose a new intelligent meeting method, WeMeet, which can automatically recommend free time and let people vote to choose the meeting time. This allows for more efficient scheduling of meetings.

## 1. Introduction

WeMeet aims to assist users in efficiently managing their meetings with time conflict-free. With this objective, WeMeet gives users the following innovative features compared to traditional meeting apps: 1. Automatic meeting time recommendation: We recommend the meeting time according to the participants' personal schedules and allow them to select the time period without time conflict. In this way, users do not have to be confused with complicated meeting time management. All they need is one click to start the meeting; 2. Majority voting determined meeting time: We allow attendees to select multiple preferred time slots and change their decisions at any time before the meeting. We will select the time with the highest number of votes as the final meeting time.

The rest of the paper introduces the architecture of the whole system, describes the corresponding API design, elaborates on essential features, and proposes further improvements.

## 2. Architecture

In our project, we utilized 8 lambda functions, 3 dynamoDB and some S3 buckets to achieve desired functionalities. The general flow and database design are the following.

**Login and Sign Up.** It triggers the *userLogin* lambda function. *userLogin* will check if the user exists in the internal network. For newly joined users, it creates a new user account with an empty calendar. If the user exists and login succeeds, we will retrieve the information for that user using *showUpcomingMeetings* lambda function.

**Create Meeting.** This is one of the most important features of our web application. It utilizes user-defined available calendars (if available), calls the *getRecommendation* and *initializeMeeting* lambda functions to generate the potentially available time slots based on all participants for this meeting, and updates all the DynamoDB tables accordingly.

**Vote Meetings.** This flow allows users to vote on the meeting time. For each meeting, each attendee will be able to select meeting time based on their own preferences. After voting, the meeting time and the related user calendars will be updated automatically. By doing so, we completely build the calendar database and make sure there is no conflict for each user and each meeting.

**Calendar System (Database Design).** We pay extra attention to avoid conflict. For each lambda function, we make sure there is no conflict for each meeting and each participant. There are some major problems with such a system. We pay special attention to the following cases: 1. Meetings may have conflicting potential times, even if there are meetings scheduled for this user. 2. For each meeting, once the scheduled time changes due to user voting, we need to update all the meetings with corresponding participants and make sure those potential times are removed except for this meeting. 3. For each user, we need to make sure the availability is exactly based on the scheduled meeting time and other available time slots. By handling those problems, we ensure our database is accurate and complete.

**Weekly Update.** We will trigger the update func-

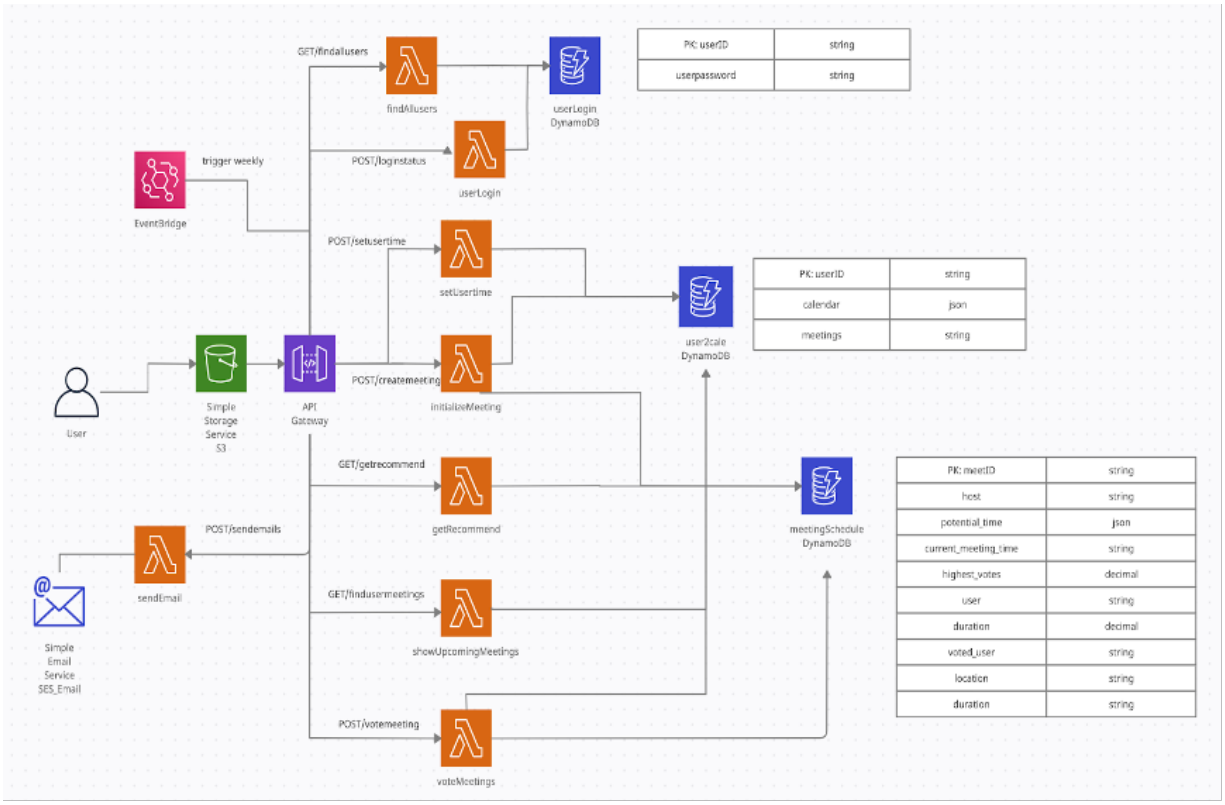


**Figure 1:** Architecture

tion to clear the weekly cache. By doing so, we clear all the information stored and make sure the database is refreshed for the new seven-day calendar. Then we complete the app processing cycle.

**Web Hosting.** In the very end, we host our web application on an S3 bucket and use SDK to access all the lambda functions and DynamoDB tables. In this way, all the users can access the web application worldwide.

## 3. API Design

**/POST/loginstatus**
Param:userID,userPassword
Methods supperted:POST
Description: for user register and login, return the status as the following: register/login/wrong password.

**/POST/setusertime**
Param:user, day, time
Methods supperted:POST
Description: for user to choose his/her available time in the next seven days.

**/POST/createmeeting**
Param:users,host,duration,location,description
Methods supperted:POST
Description: for the host to create a meeting which contains the information of participants, duration, location and meeting description.

**/GET/findallusers**
Param:None
Methods supperted:POST
Description: to show all the users of our system on the webpage.

**/GET/findusermeetings**
Param:userID
Methods supperted:POST
Description: to fine all the meetings for the user and shows their arrangement.

**/GET/getrecommend**
Param:meetID
Methods supperted:POST
Description: to find all the recommended meeting time of the meeting.

**/POST/votemeeting**
Param:user, meetID, day, time
Methods supperted:POST
Description: for the user to vote his/her preferred day and time for the meeting.

**/POST/sendemail**
Param:name, subject, message
Methods supperted:POST
Description: to send emails to the technical employees to get support of the system.

## 4. Project Details

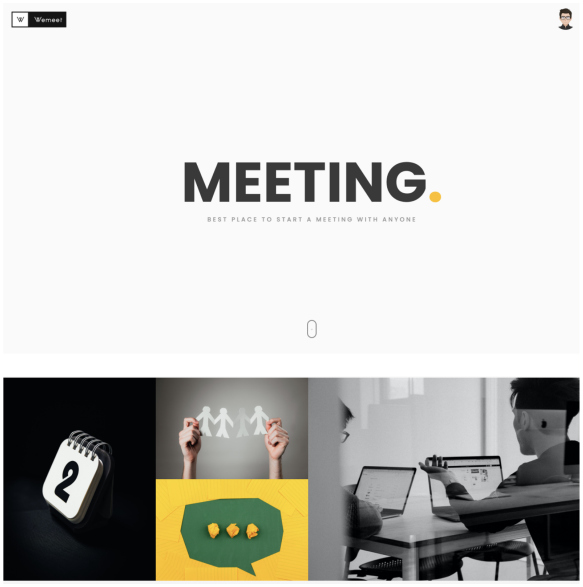


**Figure 2:** Homepage

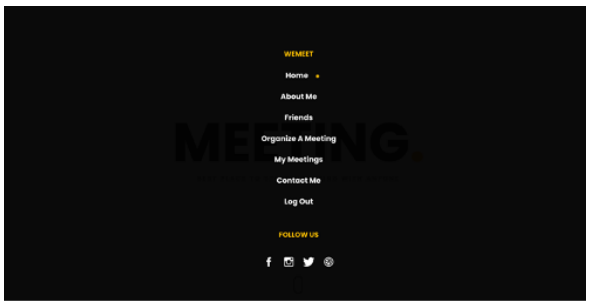


**Figure 3:** Menu

WeMeet aims to assist users to efficiently organize and manage their meetings. The system incorporates the user's personalized calendar and includes key features such as meeting organization, meeting time recommendation, timeslot voting, and making friends.

The homepage is the first page of the system, and it contains entries to all other components. The usage is detailed below. Users can return to the homepage at any moment by clicking the WeMeet logo in the upper left corner.

**Login.** Click the right upper head sculpture symbol on the homepage to go to the login page. Users must enter their username and password in order to log in. For newly joined users, the system will automatically create his/her account based on username and password inputs. Users can log out after using the system by selecting "Log Out" from the menu.

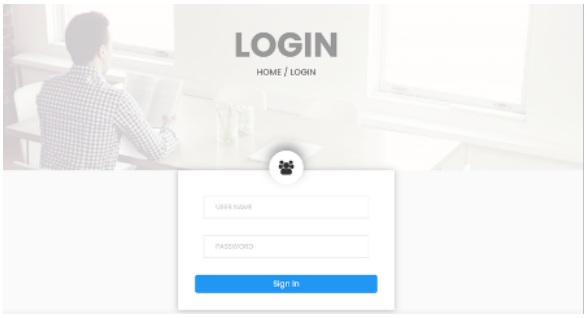


**Figure 4:** Login Page

**About me: change profile and initialize calendar.** The "About me" page displays a user's information/profile. The user could initialize his/her calendar by "choose your available slots" button. It includes all one-hour timeslots for upcoming 7 days. The meetings will then automatically satisfy all meeting attendee's available time. To improve convenience, this feature might be expanded to integrate with users' current calendars, such as Google Calendar.

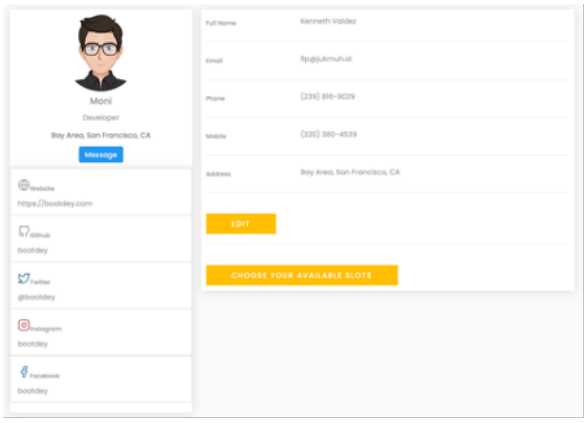


**Figure 5:** About me: change profile

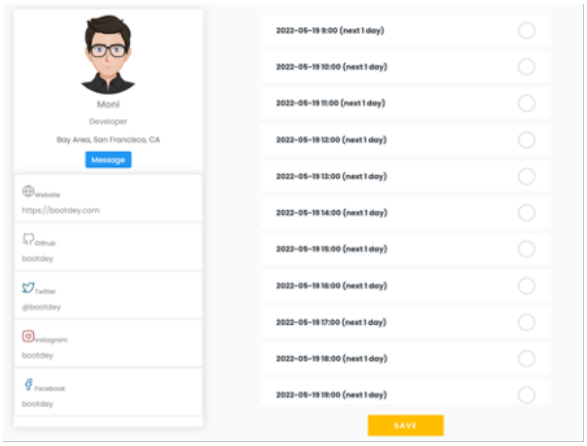


**Figure 6:** About me: initialize personal calendar

**Organize a meeting.** Anyone can organize a meeting for any others by declaring the meeting host, duration, location, description, and participants. Following the creation of the meeting, the meeting details will be displayed on each participant's meeting list, ready for their time voting.

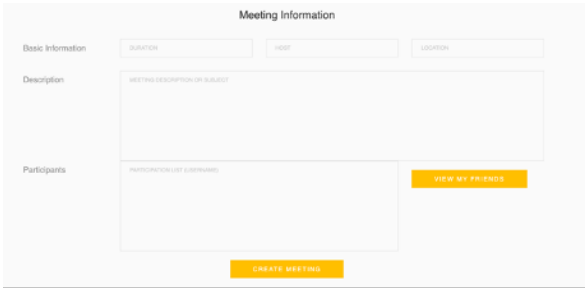


**Figure 7:** Organize a meeting

**View friends.** Friend list currently comprises every user in our system and can be served for meeting the needs of a region system, such as a company internal system or a school management system. It might also be further developed to include only real friends of our users. Users can visit this page to declare participants in a meeting.

**View my meetings.** A user's upcoming meetings are listed on the "My Meetings" tab. Once a meeting is scheduled, the system will look for timeslots that are compatible with all attendees' schedules and make recommendations.

**Meeting time recommendation and vote.** Users

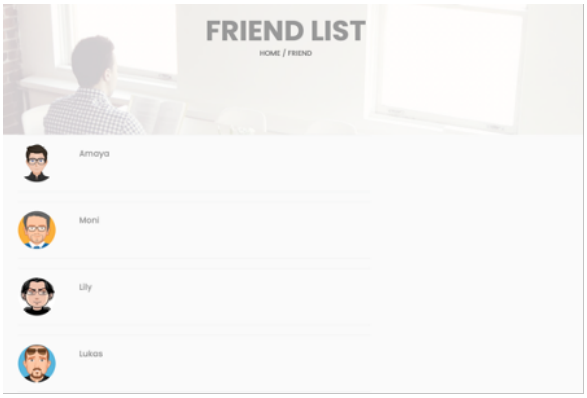


**Figure 8:** View my friends

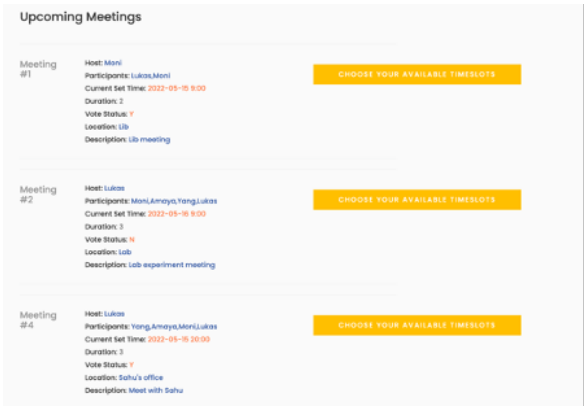


**Figure 9:** View my meetings

can vote meeting time by clicking "choose your available timeslots" button. All suggested timeslots are dynamic and depending on the schedules of all participants. It ensures that there are no conflicts with attendees' any other meetings. For example, if meeting 1 for Moni is scheduled at May $15^{th}$ 9 AM and will last two hours, then May $15^{th}$ 9 AM and 10 AM will not be recommended for Moni's other meetings and Lukas's any meetings. The "voting status" display whether a user has voted for a specific meeting. The meeting time is determined by this majority vote. If no voting happens, the system will assign it at random.

**Contact us.** We provide a way for users to report any technical issues or improvement suggestions to us. In "Contact us" page, users could fill out the reporting form and we would receive the email accordingly. We promise to follow up any feedbacks and resolve any issues.

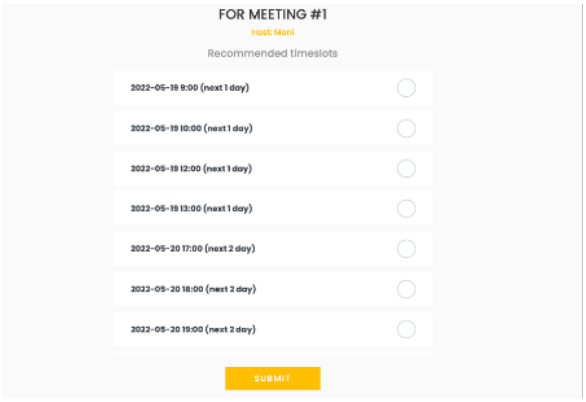


**Figure 10:** Meeting time recommendation and vote

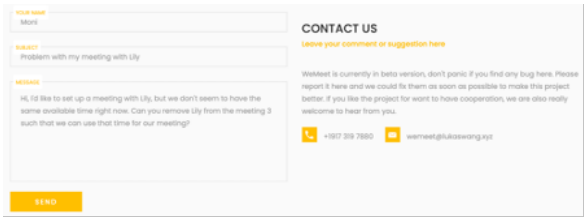


**Figure 11:** Contact us

## 5. Further Improvements

There are some aspects we expect to improve in the future:

- Integrate with user's existing calendars, ex. Google Calendar, to arrange user meetings.
- Enhance user profiles and friend pages to make it easier for people to build connections through our system.
- Create a more comprehensive notification system to inform consumers about meeting specifics.

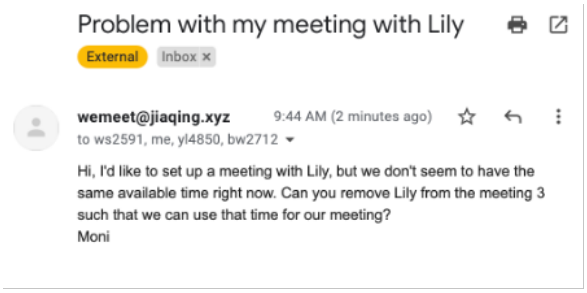


**Figure 12:** Email template

## 6. Conclusion

We hope WeMeet can be a prototype of the future meeting app. It represents an efficient and collaborative way of working. Although our functions are not perfect yet, we believe that, through future work, WeMeet will be a leading product in the market.

## Acknowledgements

This project is for COMS E6998 Cloud Computing and Big Data course at Columbia University. The system is deployed on AWS service. Thanks for professor's and TA's advice on our projects.