Progressive Web App for Mental Health

Overview:

The projects involved creating a progressive mobile web application for an already existing application and incorporate new features. The client is a highly respected and well-established player in the medical domain for Mental and Behavioral health professionals.

Client details:

Name: Confidential | **Industry:** Healthcare | **Location:** USA

Technologies:

PHP, Symfony, NodeJS, Socket.io, Mysql, Twilio, Apache, VueJS

Project Description:

The client had an existing application and wanted to incorporate new features in it. After analyzing the client's need, team@Mindfire offered to develop a mobile application. This project enabled the client to manage calendar requests, add appointments, send message etc. It was critical to have an intuitive and user-friendly UI/UX and eliminate possibilities of lag during application loading. Some salient features of the application:

- The user can view the modernized application on mobile. The application is lightweight and loads effortlessly on slower mobile networks. The team used various functionalities like Twilio, Socket.io to create the application.

- The user can easily access the list of contacts they have interacted. To create a new chat they can access the existing list or tap on "Create new". Later on, the user can select the contact they wish to send a message.

- The chat application Module helps the patient and doctors to communicate quickly. The group chat module helps the admin to send messages in the group using the group chat feature. The chatting is real-time and messages are sent to all logged-in devices to maintain proper synchronization. The chat UI is user-friendly with a smooth user experience with tools included like emoticons keyboard, automatic scrolling.

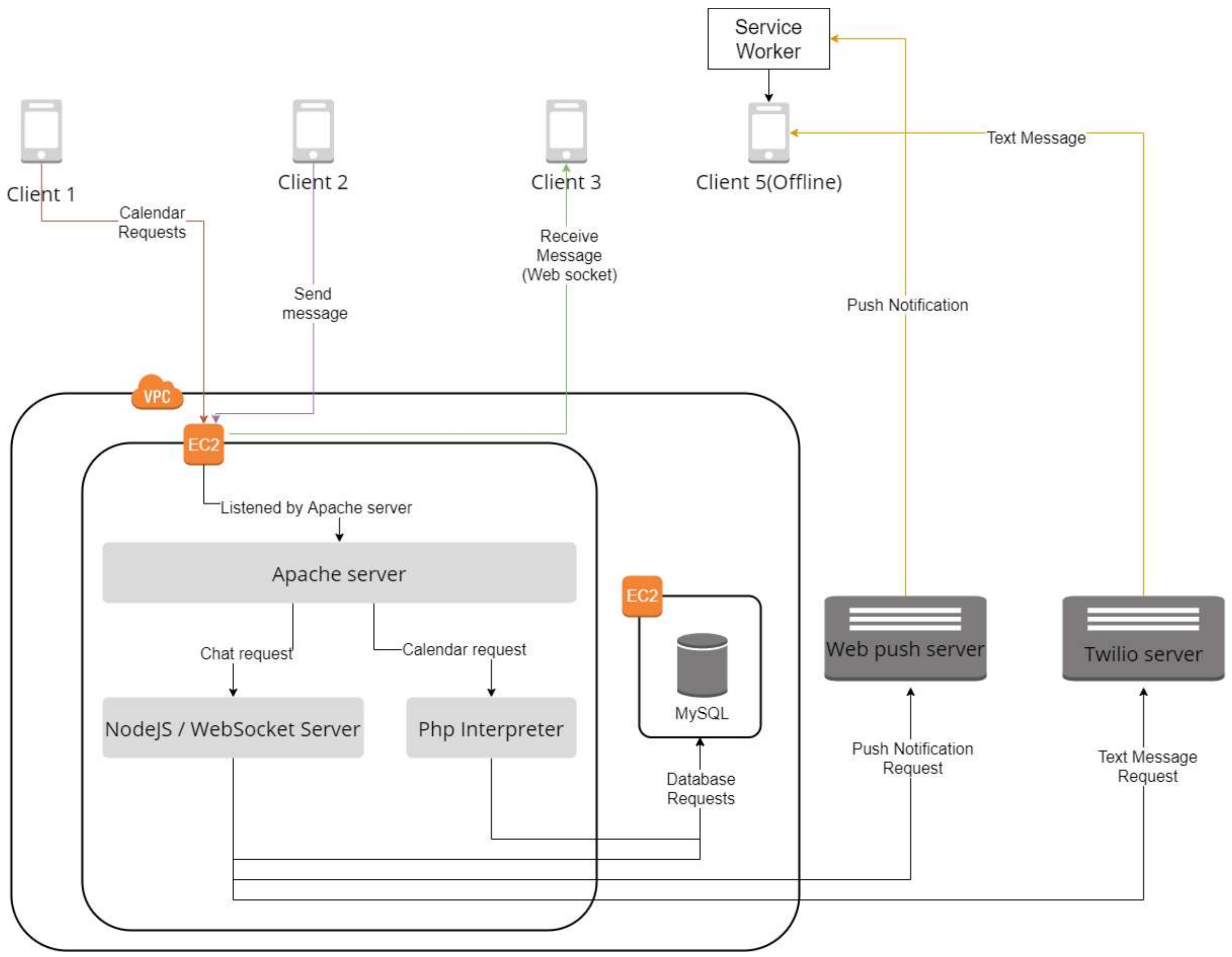
- The admin/client can add an appointment or create recurring ones (daily, weekly). The meetings can be audio or video, and the module prompts the user to select the time and date, select customized reminders and add notes.

- VueJS was chosen as the front-end library as it's lightweight and loads quickly on slower networks.

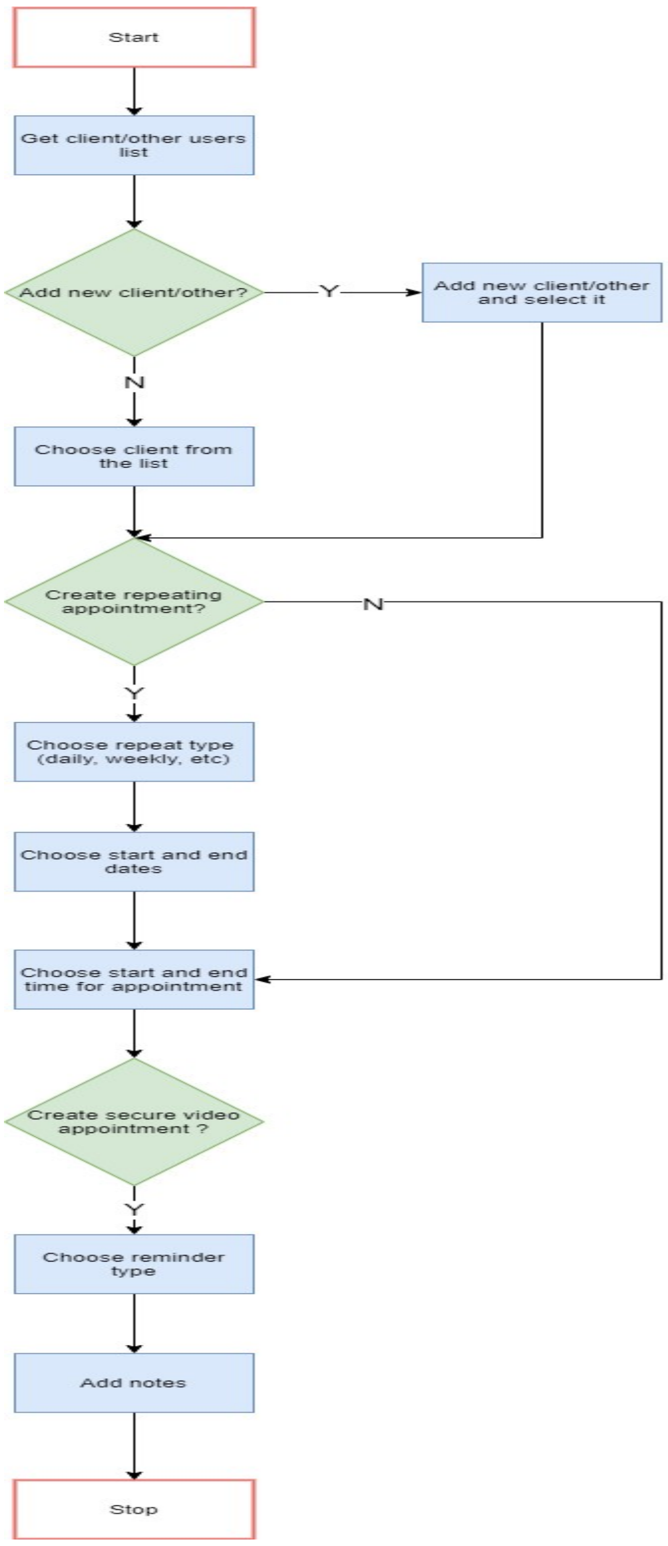
- The team used Lazy loading for static and dynamic content to reduce waiting time for each task/refresh Static content caching was done using service workers to reduce initial page load time and dynamic caching using vuex to prevent re-fetching of the same data.

- Node JS was chosen as the backend for the chat module as it's the best choice for maintaining multiple connections. Socket.io was used to enable bi-directional event-based communication and browser compatibility.

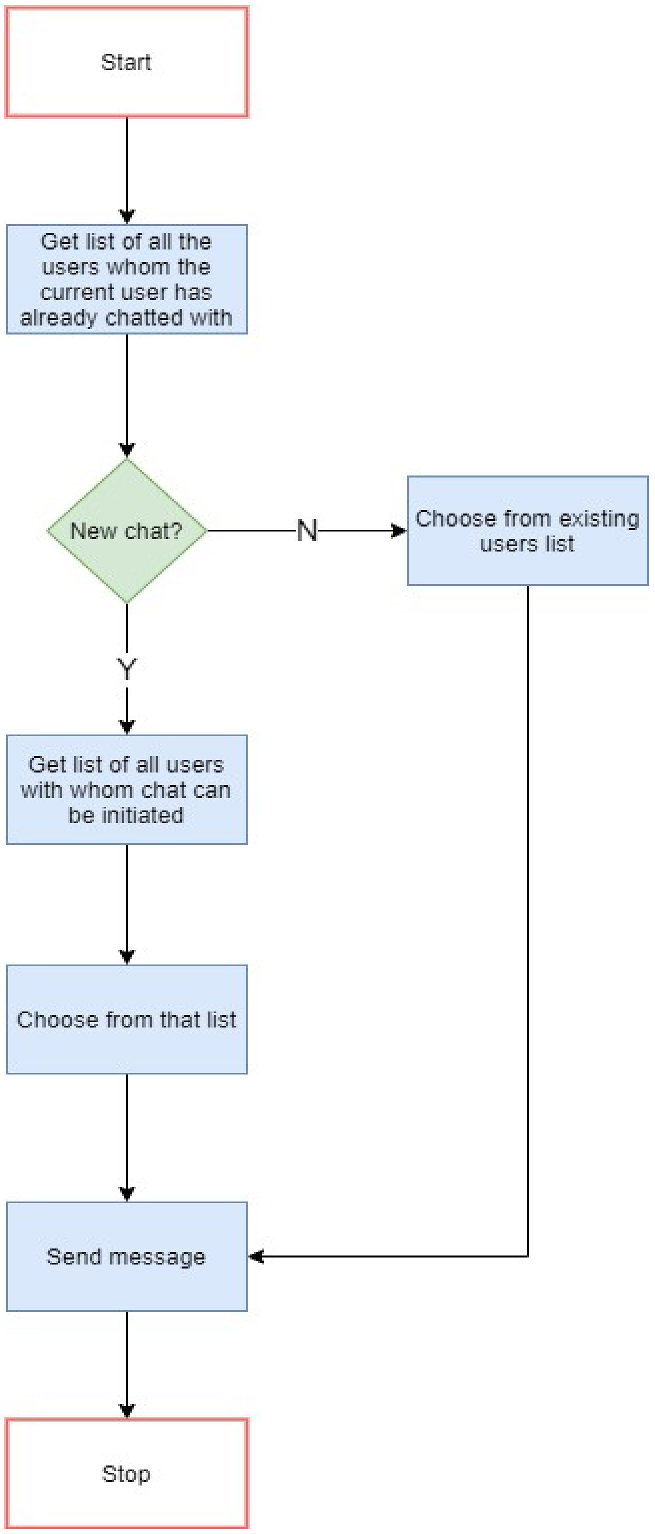
Architecture:



Workflow: Add appointment


Start
Get client/other users list
Add new client/other?
Y
Add new client/other and select it
N
Choose client from the list
Create repeating appointment?
N
Y
Choose repeat type (daily, weekly, etc)
Choose start and end dates
Choose start and end time for appointment
Create secure video appointment ?
Y
Choose reminder type
Add notes
Stop

Workflow: Send Message


Start
Get list of all the users whom the current user has already chatted with
New chat?
N
Choose from existing users list
Y
Get list of all users with whom chat can be initiated
Choose from that list
Send message
Stop