Problem statement (Original)

Covey.Town is an open-source virtual meeting application. It currently has a single room and allows any user to join the room by entering name. This causes threat to security and privacy. To overcome this issue, secured authentication can be enabled for the users where the users can register to the application and will be authorized to access Covey Room services only if they are registered.

Currently, there is no provision for the users to communicate among themselves within a room. Therefore, our idea is to implement a chat feature wherein players can communicate with each other either via private chat or by creating an open group chat. Text chat and direct messaging enhances the application by letting the users send messages to each other during the meeting.

In addition to this, the users can host events to which other users can subscribe to. The subscribed users will get instant notification regarding the event updates.

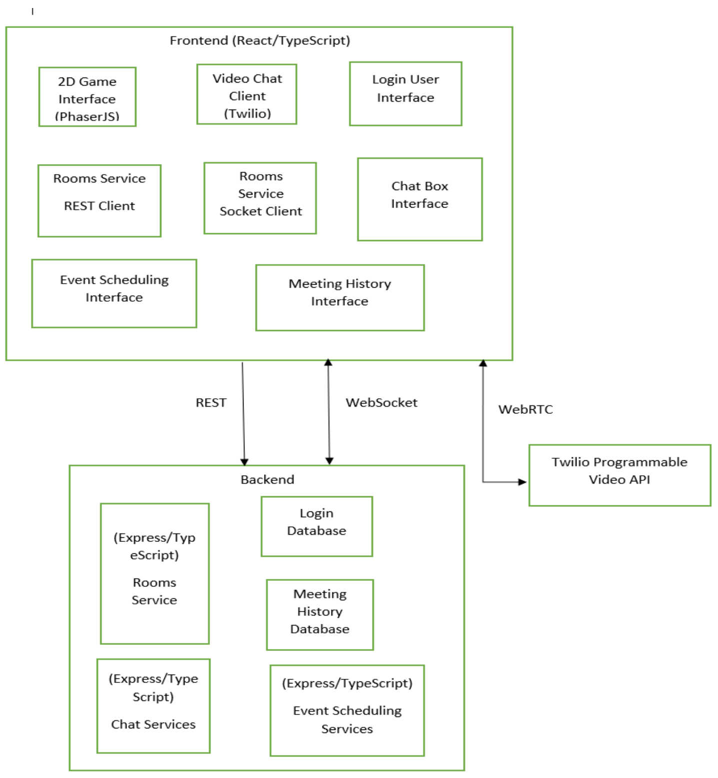
The additional features will build an interactive environment for the users on top of the video calling facility, thus improving the user experience. We find that authentication and chat are two important features that video chatting application should have. Along with that, this project would give us an opportunity to implement both the REST and Web Socket Architecture with databases into the application.

Revised User stories and acceptance

Epic	**User story**	**Acceptance Criteria**
As a user I need access to an application services securely and prevent unauthorized users from accessing the application services.	As a user, I need to register myself with the application to access a secured application service.	Ensure the user can: 1) Register to the application by providing user details. 2) Update the user details in their home page. 3) Persist the details into the database. 4) Be able to perform CRUD operations on the user.

	As a user, I need to login to the application so that I can access the history of my meetings.	Ensure the user can: 1) Login to the application by providing username and password. 2) View the history of meetings of the user.
As a user I need to be able to communicate with any user in that room through text chat facility.	As a user I need to be able to send or receive messages to/from all other users in the meeting.	Ensure the user can: 1) Send or receive messages from other users in the chat box.
	As a user I need to be able to send or receive messages to/from a user in the meeting.	Ensure the user can: 1) Send or receive messages from a user privately in the chat box.
(Optional) As a user, I need to be able to schedule an event at a particular time where other users in the room can register to this e vent and join it to have conversation.	As a user, I need to be able to create an event and let other users subscribe and join the event.	Ensure the user can: 1) Create an event by providing event details. 2) Send alerts to users subscribed to the event.
	As a user, I need to be able to join the event of other users.	Ensure the user can: 1) View list of events happening in the meeting. 2) Subscribe and get notified about the event. 3) Unsubscribe from the event.

High Level Architecture



Frontend (React/TypeScript)
2D Game Interface (PhaserJS)
Video Chat Client (Twilio)
Login User Interface
Rooms Service REST Client
Rooms Service Socket Client
Chat Box Interface
Event Scheduling Interface
Meeting History Interface
REST
WebSocket
WebRTC
Twilio Programmable Video API
Backend
(Express/TypeScript) Rooms Service
Login Database
Meeting History Database
(Express/TypeScript) Chat Services
(Express/TypeScript) Event Scheduling Services

The figure above depicts the high-level architecture of Covey.Town. The frontend client (in the frontend directory of this repository) uses the PhaserJS Game Library to create a 2D game interface, using tilemaps and sprites. The frontend implements video chat using the Twilio Programmable Video API, and that aspect of the interface relies heavily on Twilio's React Starter App. React/Typescript are used to create login, chat box, event scheduling and meeting history interface. We use REST architecture to register and login the users primarily and for few other services like retrieving meetings history and events list. We use websockets for chatting primarily.

A backend service (in the services/roomService directory) implements the application logic: tracking which "towns" are available to be joined, and the state of each of those towns. MongoDB is used as a database for Login and Meeting history. Express/Typescript are used for the event scheduling and chat services.

Work Breakdown

User Stories	Engineering	Documentation	Infrastructure
As a user, I need to register myself with the application to access a secured application service.	1) Create a registration form to get user details. 2) Display the user information in the home page of the application. 3) Store the user details into the database on registration. 4) Develop services to perform CRUD operation on the user information.	This user story serves the purpose of registering the user information into the application and store this information in the database using a registration form. 1) Open the registration form on the dashboard when we open the URL. 2) Fill in the username, email and password and confirm password details in the form and submit	1) Creation of schema on the application layer and database for storing the user details. MongoDB will be used as a database.
As a user, I need to login to the application so that I can access the history of my meetings.	1) Create a login page to let the user enter username and password. 2) Retrieve the history of meetings of the user and display it in the home page of the application.	This user story enables the user to login to the application and access the application services securely. 1) Open the login page on the dashboard. 2) Fill in the username and password and submit. 3) It should redirect you to the homepage	1) Authentication and authorization service for verifying the user credentials. 2) Creation of schema on the application layer and database for storing the meeting history details for every user. MongoDB will be used as a database.
As a user I need to be able to send or receive messages to/from all other users in the meeting.	1) Create a global chat box interface and communicating via client server web sockets among users to provides the facility to broadcast messages to everyone in the room.	This user story enables the user to chat with other where they can send and receive messages with other users in the room. 1) Click on chat button at the bottom of the room, it opens a modal window	No separate infrastructure required beside using web sockets.

		2) Select the drop down to select if the chat is global or private. 3) Select global option and then start typing the chat messages at the text field provided	
As a user I need to be able to send or receive messages to/from certain number of users in the meeting.	1) Create a private chat box for two or more users to communicate via client server web sockets.	1) Click on chat button at the bottom of the room, it opens a modal window 2) Select the drop down to select if the chat is global or private. 3) Select private option and then select the user to chat with and then start typing the chat messages at the text field provided	No separate infrastructure required beside using web sockets.
As a user, I need to be able to create an event and let other users subscribe and join the event.	1) Develop events with an event front end interface and scheduling by providing event details and the timing to all other users in the room. 2) Send alerts/notifications to users subscribed to the event few minutes before the event.	This user story enables the user to create and host events along with the event details and the time of the event. 1) So, click on the host event button, to create an event where the user can give the event details and then submit the event request with the time details. 2) This event alert is then sent to all the other users in the room along with the event information and the time of the event for them to subscribe to it.	No separate infrastructure required beside using web sockets.

As a user, I need to be able to join the event of other users.	1) List the events happening in the meeting with the event details and timings. 2) Provide option to Subscribe to events that doesn’t have time conflicts and receive notification about the event. 3) Provide option to Unsubscribe from the event at any time before the event starts.	This user story enables the user to join to the available unexpired events to the room. They can subscribe to the events that they want to join. 1) Select the available events panel and select an event from the list of selected events. 2) Subscribe to the selected event by clicking on subscribe. 3) Subscribed events should appear under upcoming events where the users can unsubscribe to the events if they want to unsubscribe from the event.	No separate infrastructure required beside using web sockets.

Schedule

Week 1 to Week 5 -> March 15 to April 15

Tasks	Timeline	Name
Create a registration form to get user details.	Week 1	**Front-end:** Ram Tarun Balagam and Mownika Asokan **Back-end:** Setting up the environment: Ram Tarun Balagam Registration and Login Services: Satyanarayana Vadlamani Meeting history Services: Mownika Asokan **Testing:** All
Display the user information in the home page of the application.	Week 1	
Store the user details into the database on registration.	Week 1	
Develop services to perform CRUD operation on the user information.	Week 1	
Create a login page to let the user enter username and password.	Week 1	
Retrieve the history of meetings of the user and display it in the home page of the application.	Week 1	
Testing and Documentation - Epic 1	Week 1	
Create a global chat box interface and communicating via client server web sockets among users to provides the facility to broadcast messages to everyone in the room.	Week 2 and Week 3	**Front-end:** Ram and Mownika **Back-end:** All **Testing:** All
Create a private chat box for two or more users to communicate via client server web sockets.	Week 3	
Testing and Documentation - Epic 2	Week 3	
End to End testing & Buffer	Week 4	**Testing:** All
Develop events with an event front end interface and scheduling by providing event details and the timing to all other users in the room.	Week 4*	Ram
Send alerts/notifications to users subscribed to the event few minutes before the event.	Week 4*	Satya

List the events happening in the meeting with the event details and timings.	Week 4*	Mownika
Provide option to Subscribe to events that doesn't have time conflicts and receive notification about the event.	Week 4*	Ram
Provide option to Unsubscribe from the event at any time before the event starts.	Week 4*	Mownika
Testing and Documentation - Epic 3	Week 4* and Week 5*	Satya
Video demonstration	Week 5	All

- **Note:** Epic 3 is an ambitious task which would be developed after intime completion of the previous epic's according to the mentioned timeline.