Synchronous

Team 10 Design Document

Kevin Wu, Ryan Borzello, Saksham Jain, Shreyas Iyengar

Table of Contents

Purpose

Synchronous is positioned as a real time collaboration tool, with the ability to create virtual workspaces for teams, and offering tools and services inside the workspace that will allow for better communication and collaboration, such as a virtual whiteboard, notepad as well as file sharing, all without having to set up an account or register with the website. We envisioned it as bridging the gap between Zoom and the Google suite of applications, bringing security as well as flexibility all without the overhead of storing information or creating an account.

Collaborating with teammates on small projects is commonplace in college and the workforce, yet setting up a workspace for the team with existing tools is cumbersome. To set up a Zoom meeting, one needs an account, and there is a limited amount of minutes one would have for a free meeting, if not affiliated with a company or organization. Zoom also doesn't have tools for drawing or typing, and is primarily a video conferencing app.

Google Drive, on the other hand, has the ability to type, chat, etc. with the added bonus of being able to see the changes happen in real time. However, the overhead of having to share the documents with each team member, and quick collaborative notepad services don't have other useful features such as chat, file sharing, or whiteboards. Synchronous puts all these features in one place allowing teams to start working together as quickly as possible. With Synchronous, users can set up a workspace with a custom URL, and other members can easily join by going to the URL. With Synchronous, you get real-time collaboration without the need for creating an account or sending out invitations. We envisioned this as a really fast, super light program dedicated for quick team meetings or minimal conferences.

Functional requirements

1. Workspace initialization
    - 1.1. As a team leader, I would like to be able to create a workspace.
    - 1.2. As a team leader, I would like to set a workspace's unique ID.
    - 1.3. As a team leader, I would like to be able to password protect my workspace during creation so that unauthorized users cannot access it.
    - 1.4. As a team leader, I would like to optionally let others view my workspace without a password but have editing privileges be password-protected so I can share something for others to see but not edit.

2. Workspace access
    - 2.1. As a team member, I would like to be able to access a workspace through its URL.
    - 2.2. As a team member, I would like to be able to access a workspace by entering its unique ID.
    - 2.3. As a team member, I would like to be able to send an email to invite people.
    - 2.4. As a user, I would like to be able to change my nickname in my workspace so that other users can see who is making changes.
    - 2.5. As a user, I would like to be able to see who else is online in my workspace.

3. Text editor app
    - 3.1. As a user, I would like to view a list of previous workspaces that I have opened.
    - 3.2. As a user, I would like to be able to add a text document app to my workspace that synchronizes the text with all other users and shows what text I am selecting to them.
    - 3.3. As a user, I would like to be able to view and revert to a point in the history of a text document (if time allows).
    - 3.4. As a user, I would like to be able to create rich text content in a text document app.
    - 3.5. As a user, I would like to add suggestions to sections of text in a text document app.
    - 3.6. As a user, I would like to create a text document app by uploading a PDF document, an Office Open XML document (.docx), or a plain text file and be able to edit the document's contents (if time allows).
    - 3.7. As a user, I would like to be able to download the contents of a text document app as a PDF document, an Office Open XML document (.docx), and a plain text file.

4. Whiteboard app
    - 4.1. As a user, I would like to be able to add a whiteboard app to my workspace that synchronizes all changes on the whiteboard.
    - 4.2. As a user, I would like to draw on the whiteboard with multiple colors.
    - 4.3. As a user, I would like to use different marker sizes for drawing on the whiteboard.

- 4.4. As a user, I would like to be able to erase strokes on the whiteboard.
- 4.5. As a user, I would like to see other people's pointers in a whiteboard app.
- 4.6. As a user, I would like to be able to upload an image to the whiteboard app so that I can draw on top of it.
- 4.7. As a user, I would like to be able to download the contents of a whiteboard app as a PDF document, a JPG file, or a PNG file.

5. Other apps
   - 5.1. As a user, I would like to be able to add a chat app to my workspace so that I can communicate with other users connected to the workspace.
   - 5.2. As a user, I would like to be able to add a file sharing app to my workspace so that other users connected to the workspace can download the file I am sharing.

6. Workspace management
   - 6.1. As a user, I would like to be able to share any type of file in the workspace.
   - 6.2. As a user, I would like to be able to minimize, rearrange, and resize an app without affecting what other users connected to the workspace see so that I can customize my workspace layout.
   - 6.3. As a user, I would like to be able to view what apps are minimized in the workspace in a sidebar so I can reopen them.
   - 6.4. As a user, I would like to be able to delete an app so that I can discard its contents.
   - 6.5. As a user, I would like to download the contents of all apps at once (if time allows).
   - 6.6. As a user, I would like to upload the contents of all apps to recreate a past workspace (if time allows).
   - 6.7. As a user, I would like to split the workspace into multiple workspaces within the same session.
   - 6.8. As a user, I would like to be able to edit data in apps if my connection is lost and have my changes be synced when I rejoin the connection.

7. Support
   - 7.1. As a user, I would like to be able to submit bug fixes and user issues to the developer.
   - 7.2. As a new user, I would like to have a tutorial explaining all the features when using the app for the first time.

Non-functional requirements

1. Hosting
    1.1. As a developer, I would like the clients to connect to the database to synchronize changes in the apps in real time.
    1.2. As a user, I would like to be able to create a workspace within 200ms.

2. Security
    2.1. As a user, I would like my workspace IDs to be guaranteed to be unique.
    2.2. As a user, I would like to have any communication between the client and the server as well as the client and the database be encrypted.

3. Usability
    3.1. As a user, I would like to access Synchronous on any web browser, including browsers on mobile devices.
    3.2. As a user, I would like the user interface to be intuitive/easy to understand.

4. Performance
    4.1. As a developer, I would like new updates to the database to be shared to all clients within 500ms.

5. Resilience
    5.1. As a developer, I would like each of the server's apps to handle at least 1,000 concurrent connections.
    5.2. As a developer, I would like the main server (serving static files) to handle at least 3,000 concurrent connections.

Design Outline

High Level Overview

This project will be a web project that will allow any individual, regardless of system, to be able collaborate and use the workspace without any hassle of setup.

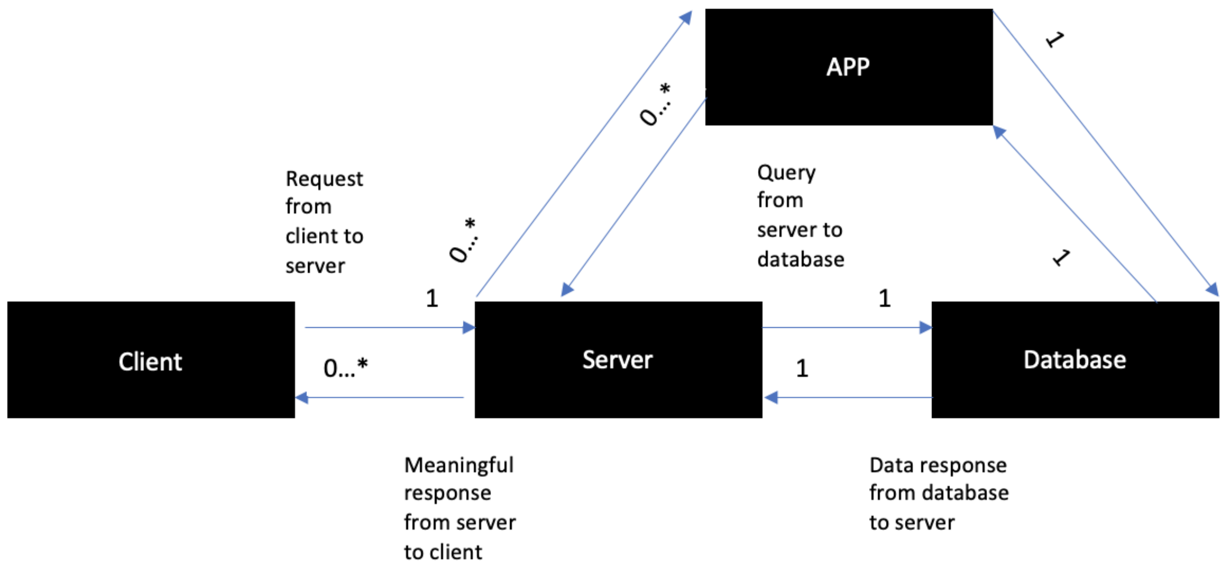
The main user who sets up the meeting will have to go online to our website and request a new workspace. They will have the option of entering their email before entering the workspace, which will allow them to keep a copy of the workspace documents after the meeting is over. No registration is required.

The workspace itself will be a full screen app with individual applets containing tools that users can drag and drop into the workspace. They will be able to drop as many tools as they can fit on the screen, and can also drop many tools of the same kind, in the event that someone wants multiple white boards or multiple notepads. The individual windows will also be resizable to fit the users needs.

The workspace will have a unique ID assigned to it each time a workspace is generated, and this workspace will be available for 24 hours, accessible by that unique ID. No other person will be able to create a new workspace with that unique ID for as long as it is still in use. Users will also have the ability to set an access code to prevent people from joining even if they have the meeting ID.

Users will also be able to create as many workspaces as they need, especially if they are working on multiple projects.

Components


APP
Client
Server
Database
Request from client to server
Query from server to database
Meaningful response from server to client
Data response from database to server
1
0...*
0...*
0...*
1
1
1
1

1. Client
    a. The client will be a web-based workspace interface written in React.
    b. The client will send a workspace creation request to the server, along with a given email.
    c. Application functionality will be delivered through open-source web applications embedded in the client that are hosted by Synchronous.

2. Server
    a. The server receives workspace creation requests and ensures that the created workspaces have unique IDs.
    b. The server will query the database to retrieve existing workspaces.

3. Database
    a. The main database will be a relational database that stores data about all workspaces, such as the apps that have been created and the users that are viewing the database.
    b. The database for each app will be independent from the main database, allowing drop-in replacement apps for a different text editing or drawing experience.

4. App
    a. The apps will be displayed in the Client, showing its contents to the user and allowing them to edit them.
    b. The apps will be based on open-source software (e.g. Firepad), with modifications to customize it to suit Synchronous' needs.

Design Issues

Functional Issues

1. Do users need to log in to use our service?
    - Option 1: Users must create an account using an email/password
    - Option 2: In addition to creating an account, allow users to log in with their existing accounts on other services such as Google using OAuth 2.0
    - Option 3: No creation of accounts

    Choice: Option 1

Justification: Creating an account would allow for features that rely on identification of users, such as setting an owner of a workspace and giving certain privileges only to the workspace's creator. However, forcing users to log in, even through their accounts on other services, would increase the time and clicks the user has to go through to create a workspace. Therefore, to differentiate Synchronous from existing services, we will allow users to access all features without logging in.

2. How should users access their workspaces?
    - Option 1: Allow users to create their own unique workspace ID with password protection optional
    - Option 2: Auto-generate unique workspace IDs with password protection optional

    Choice: Option 2

Justification: Auto-generating a unique workspace ID will allow for quick access to the workspace, as well as greater security to prevent access by unwanted people. This will also reduce conflicts that may occur by many people choosing the same workspace ID for their workspace. A randomly generated workspace ID, in the same vein as zoom will allow for reduced collisions, and reduce delays in accessing a workspace.

3. How long should workspaces be saved?
    - Option 1: Save workspaces indefinitely
    - Option 2: Allow users to set a time before expiry
    - Option 3: Delete workspaces automatically after 24 hours

Choice: Option 3

Justification: Saving workspaces indefinitely will defeat the purpose of the app, as the app is meant for quick collaboration, without the overhead of an account. One could save a workspace without an account as long as they have the specific meeting ID and password, but without an account, the user won't be able to find any way of accessing that workspace if they lose the unique workspace ID that is auto generated by the app. Allowing users to set a time before expiry does make sense, but if some people choose to set it to be available indefinitely, it would again fall into the same category as option 1. There is also the risk of overload if millions of users don't free up their workspaces when they're done with them, or decide to keep them going for a long time. Option 3 is the best choice to provide a decent amount of access to the user, while avoiding major stress or overhead on the system.

4. How should the apps be organized in a workspace?
    - Option 1: Have the apps be listed in a sidebar, allowing users to drag and drop them into their workspaces and reorganized like desktop windows
    - Option 2: Same as option 1 but have the location and size of windows be synchronized across all viewers of the workspace
    - Option 3: Have the open apps be openable one at a time in the main workspace window

Choice: Option 1

Justification: Option 2 wouldn't work because people will be working across different platforms with different viewing styles in mind, and keeping everyone to the same layout wouldn't make sense, especially if a person would want the drawing to be more visible, and another person would want the text to be more visible on their screen. Different strokes for different people. Option 3 would just lead to more clutter, and we want to maximize the space as much as possible for the workspace.

5. How long should files shared by file sharing persist?
    - Option 1: Use peer-to-peer file sharing, when the sharer disconnects the option to download the file disappears
    - Option 2: Upload the file to Synchronous servers, delete the file when the workspace is deleted

- Option 3: Same as option 2 but also allow anybody to delete the file from Synchronous servers at any time

Choice: Option 3

Justification: Option 1 isn't feasible as it might saturate the upload bandwidth of the uploading user if multiple users are downloading the file. Furthermore, if the uploader accidentally disconnects from the workspace while others are downloading the file, download progress would be interrupted and have to be restarted once the uploader rejoins. Option 2 is limited if we cap the maximum file size or number of files that can be shared at the same time. If we have option 3, we can clear files before adding more if needed.

Non-functional Issues

1. What language/framework should be used for the back-end of Synchronous?
    a. Option 1: Node.js with Express.js
    b. Option 2: Python with Django
    c. Option 3: PHP

Choice: Option 2

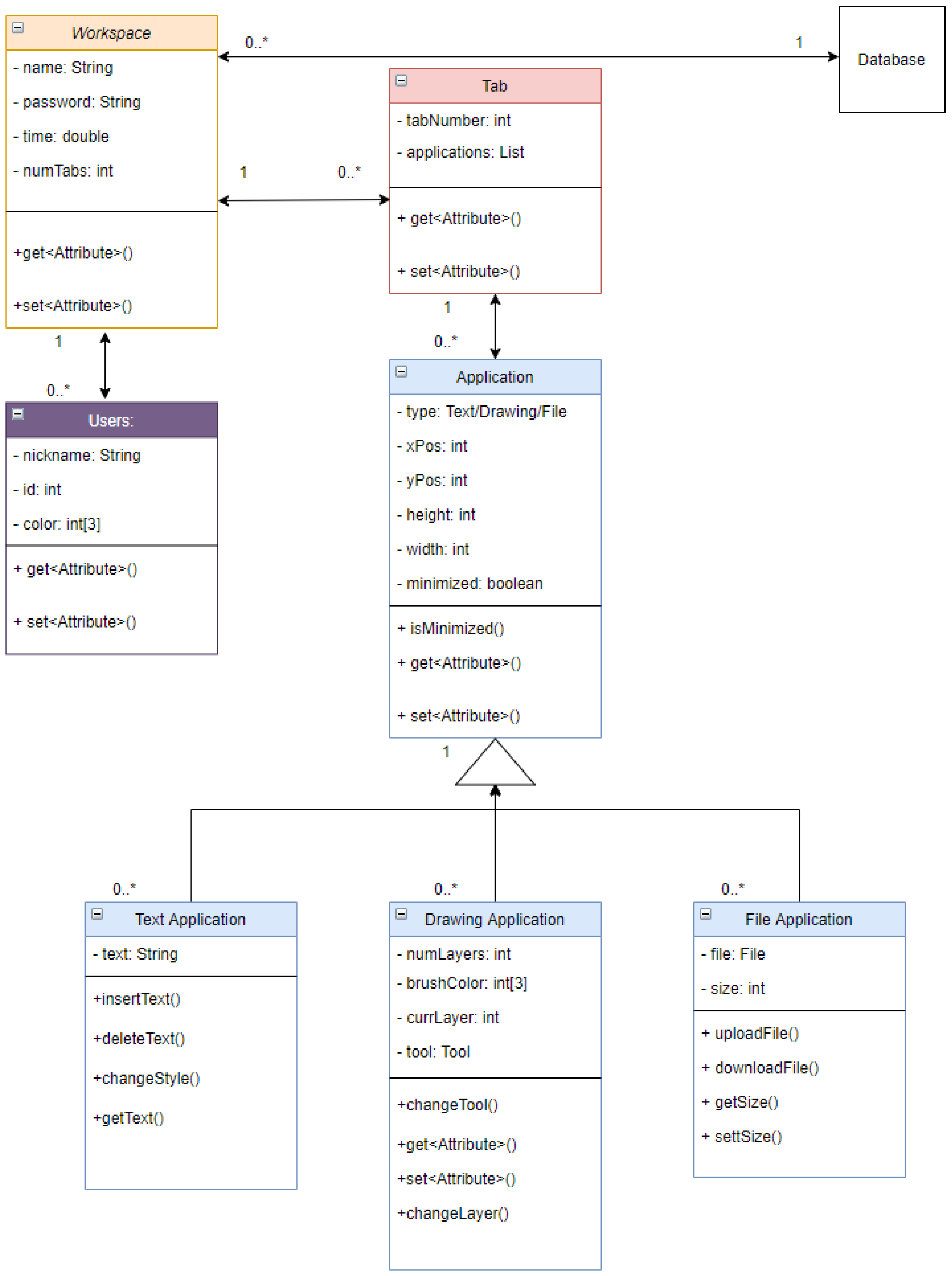
Justification: Django has an extensive existing user base that makes it easy for new users to jump in and learn. Many of our team members are familiar with Python, so we do not have to learn JavaScript or PHP to create the backend. Lastly, Django's models will make implementing our database models easy as well as abstract away database queries.

2. How do we synchronize user changes and ensure that all users see the same document without utilizing too much bandwidth?
    a. Option 1: Use the ShareDB real-time database backend to ensure that conflicting edits are resolved and data is synchronized
    b. Option 2: Embed pre-existing open source collaborative text editors and whiteboards

Option 2: We didn't want to reinvent the wheel, so we decided to embed the tools we need in our app and make sure they mesh well together.

Design Details

Class Design



Class Descriptions

- <u>Workspace</u>
  - Workspace object is created when a User creates a new workspace.
  - The workspace will be the main class that all others ultimately end up interacting with.
  - Contains a name, password, time (time of creation), and numTabs (the number of tabs the workspace has).

- <u>User</u>
  - User object is created when a new User joins a workspace.
  - There can be multiple users for a workspace.
  - Users will input their nickname, which will be the name that will be presented to other online users.
  - The user's id and color will be randomly generated when the user object is created.
  - The displayed name of the user will be stored in the nickname field.
  - The id will be used to uniquely identify the actions of each user.

- <u>Tab</u>
  - Users have the options to create multiple tabs, each with a fresh version of the workspace, allowing users to separate ideas or create more space for the existing project.
  - Each tab will have the same amount of options as the first instance of the workspace, with options to rename as well as organize.
  - The tabs will all be stored under the same workspace id for the duration of the workspace availability.

- <u>Application</u>
  - Application is an abstract class that other applications will extend from.
  - Each tab can contain multiple Applications.
  - Each Application has a type, which indicates which type of application it is.
  - The position (xPos, yPos), size (height, width), and whether or not it is minimized on the user's screen will be stored as well.

- <u>TextApplication</u>
  - TextApplication is a subclass of Application class.

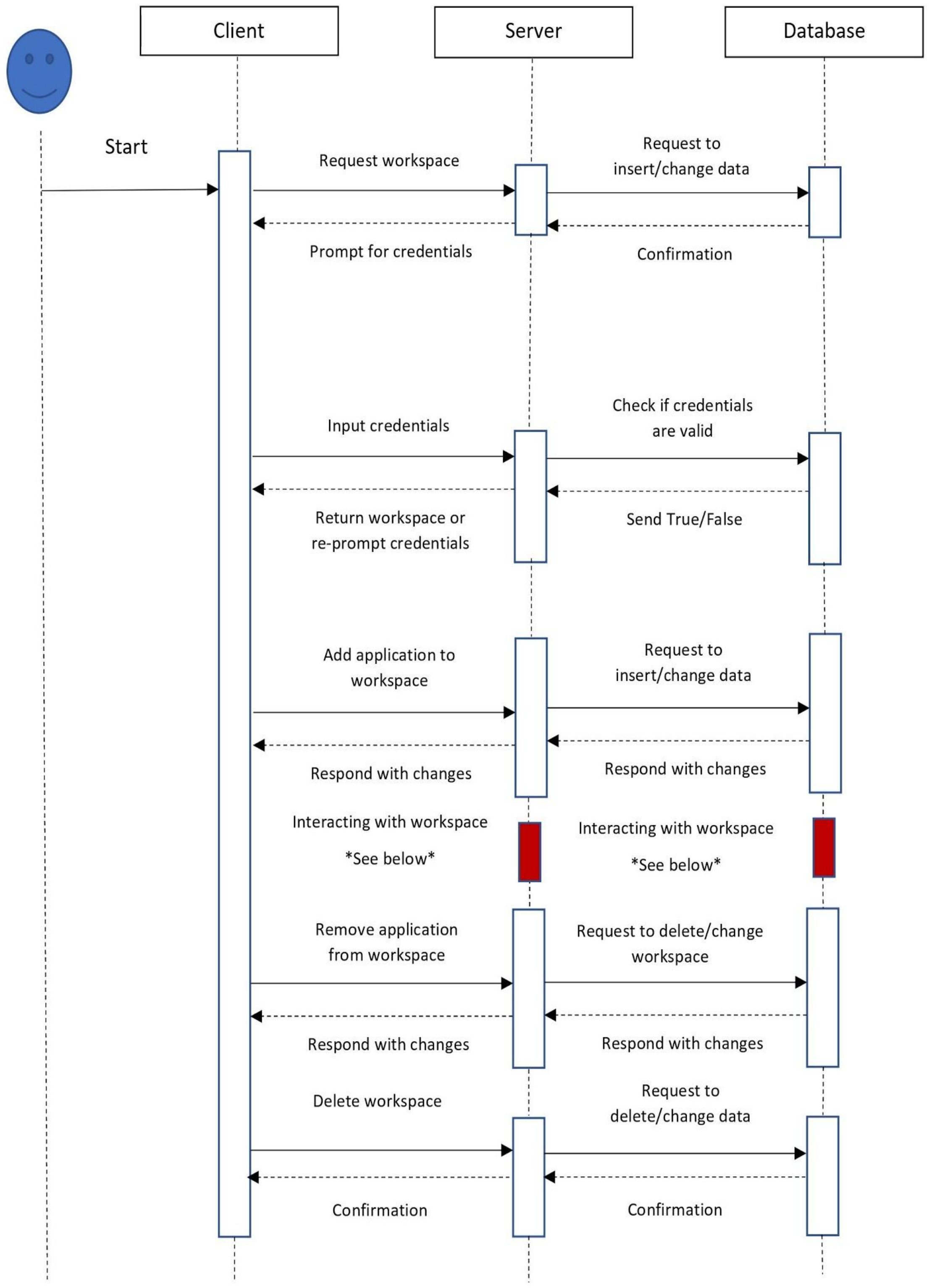
  - TextApplication is created when user drags it out.
  - The TextApplication will allow for typing and editing of text.
  - Contains the text that the users are editing.
  - Methods will allow for users to insert, delete, and style text.

- <u>DrawingApplication</u>
  - DrawingApplication is a subclass of Application class.
  - DrawingApplication is created when user drags it out.
  - The DrawingApplication will allow for collaborative drawing on screen by all users.
  - Contains number of layers, color of the brush, current layer number and tool.
  - The methods will allow users to change their tools and layers.

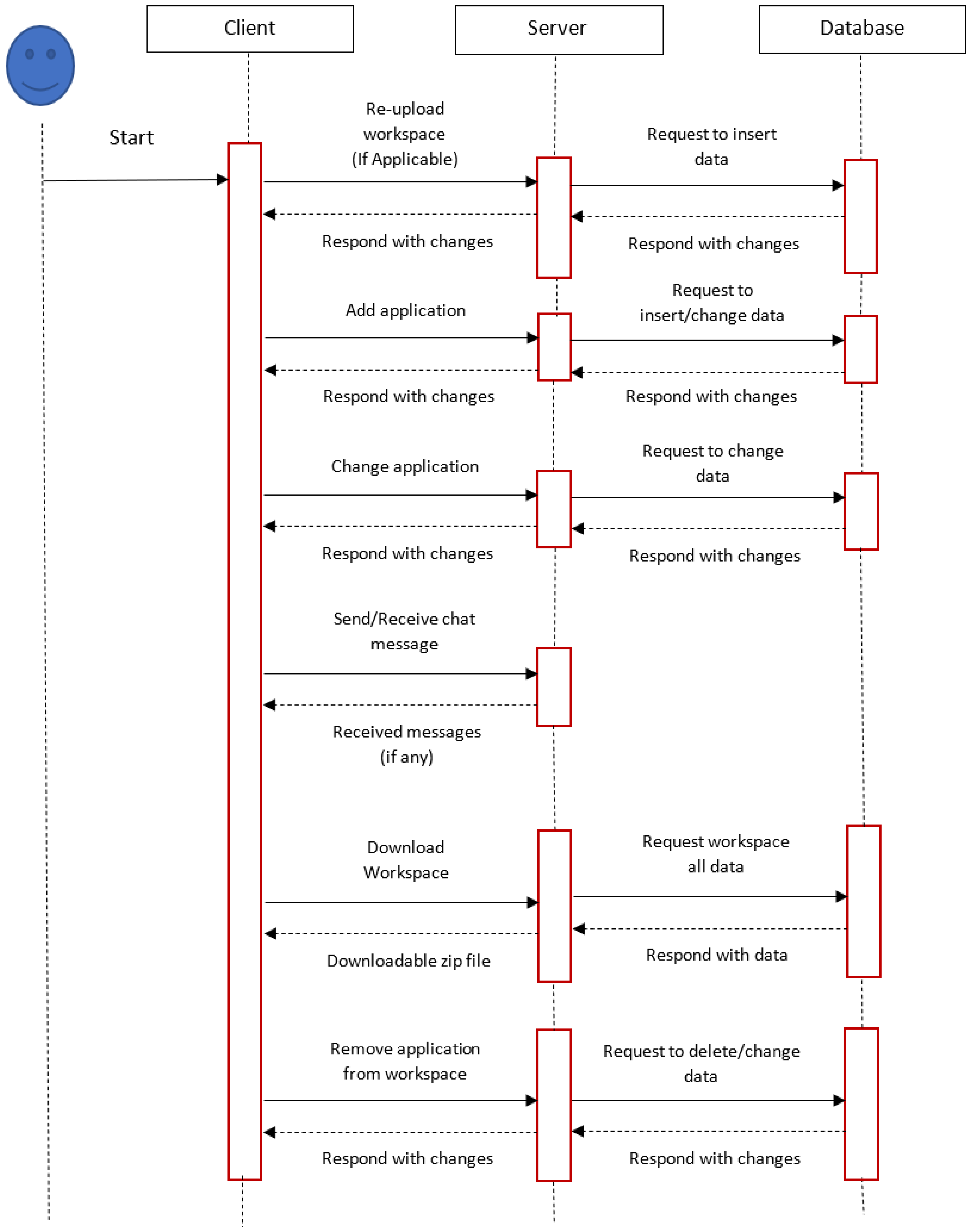
- <u>FileApplication</u>
  - FileApplication is a subclass of Application class.
  - FileApplication is created when user drags it out.
  - The FileApplication will allow for upload and download of files.
  - Contains the uploaded file and the size of the file.
  - The methods allow for file upload and download.

Event Sequence Overview

Creating/deleting a workspace:

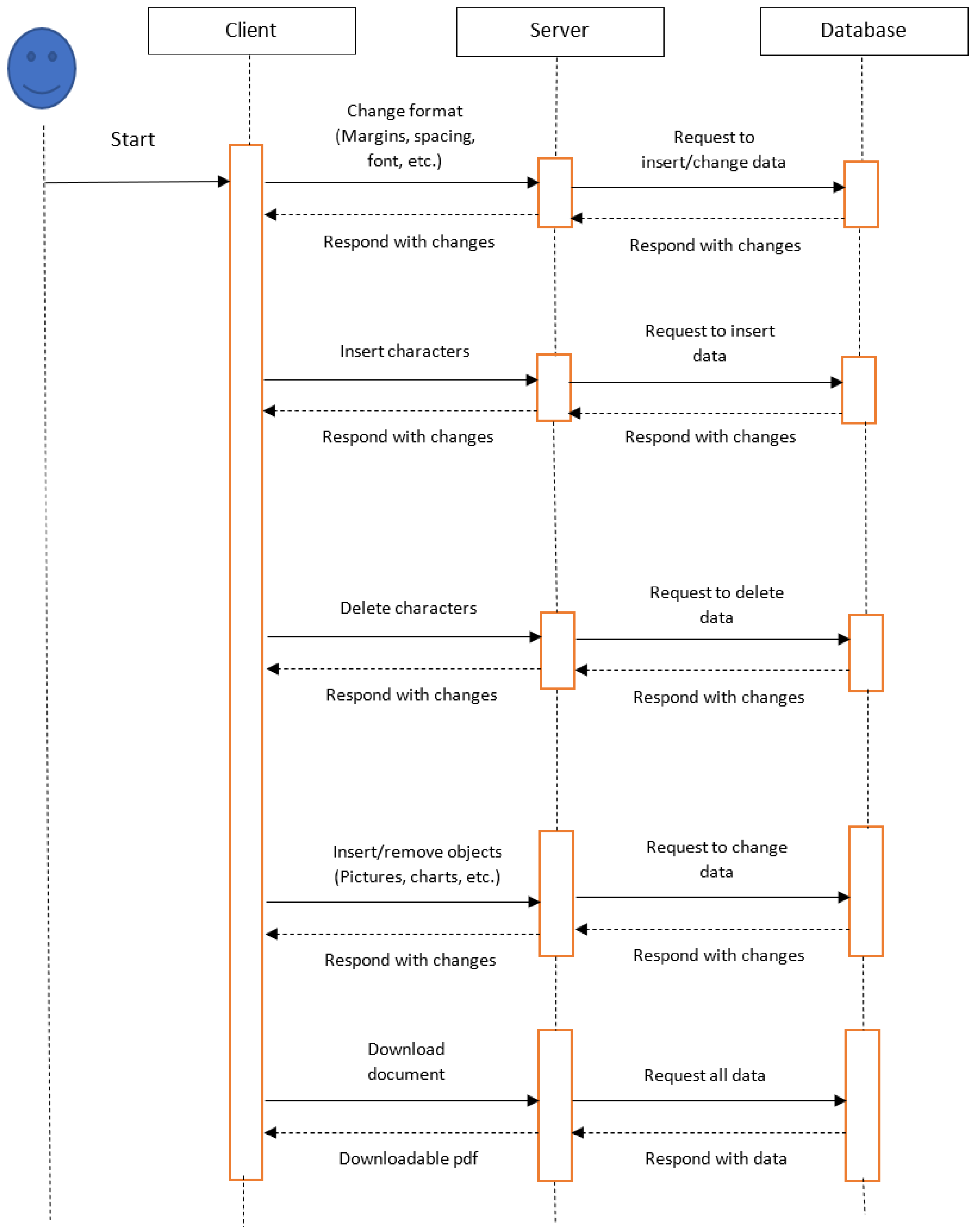

Client
Server
Database
Start
Request workspace
Request to insert/change data
Prompt for credentials
Confirmation
Input credentials
Check if credentials are valid
Return workspace or re-prompt credentials
Send True/False
Add application to workspace
Request to insert/change data
Respond with changes
Respond with changes
Interacting with workspace *See below*
Interacting with workspace *See below*
Remove application from workspace
Request to delete/change workspace
Respond with changes
Respond with changes
Delete workspace
Request to delete/change data
Confirmation
Confirmation

Interacting with workspace: 🟥



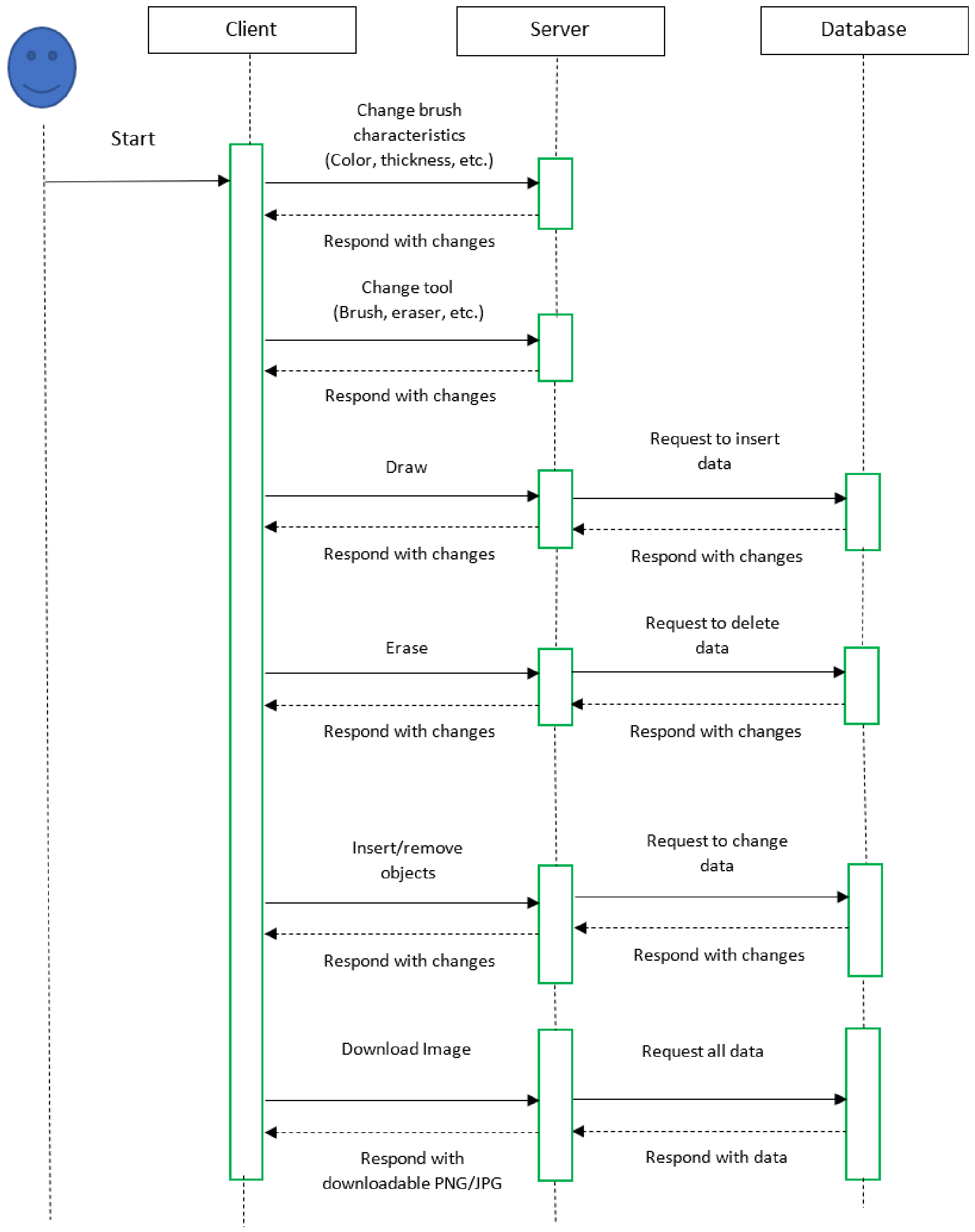
Text Application Sequence Diagram

Using the text application:



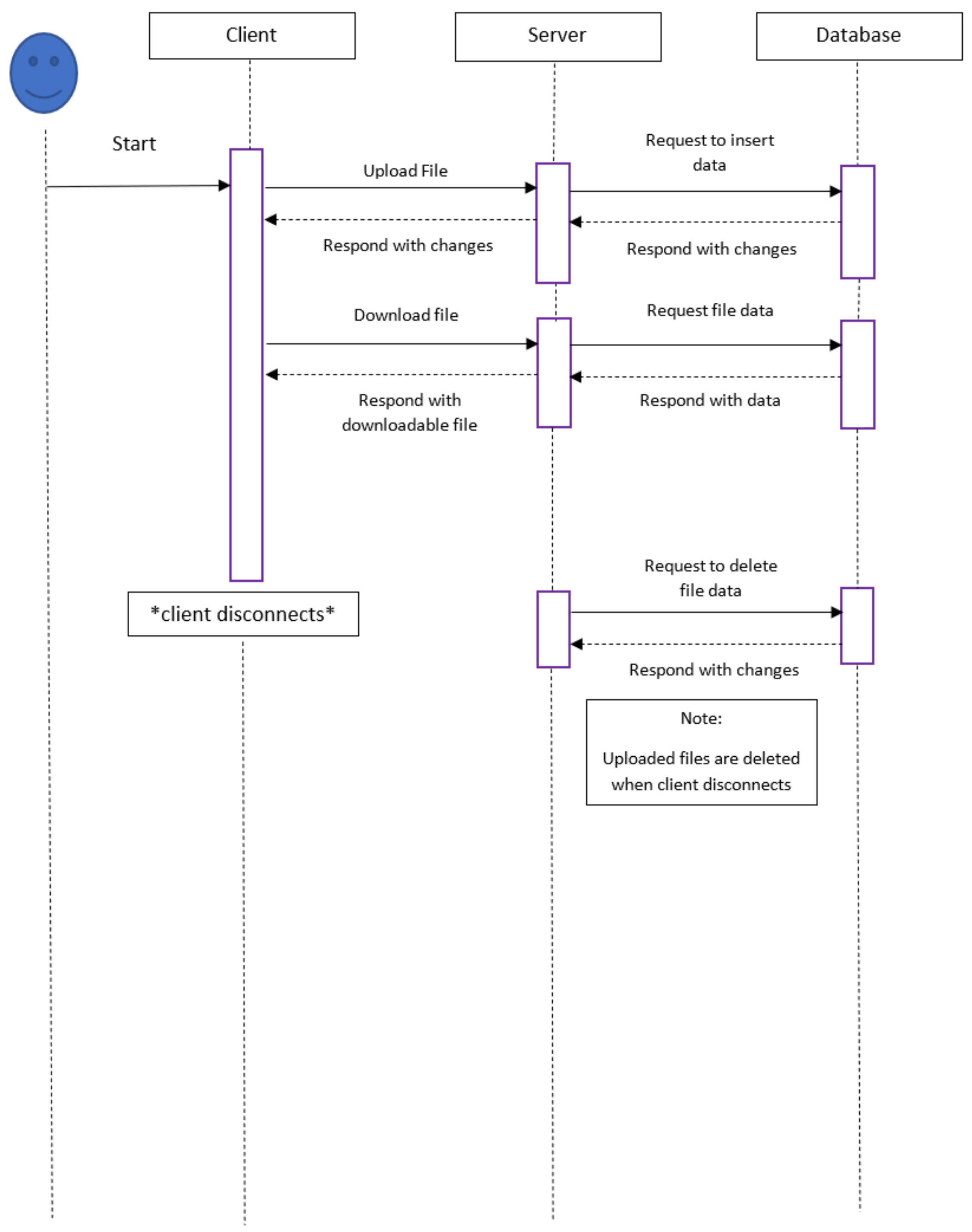
Drawing Application Sequence Diagram

Using the drawing application:

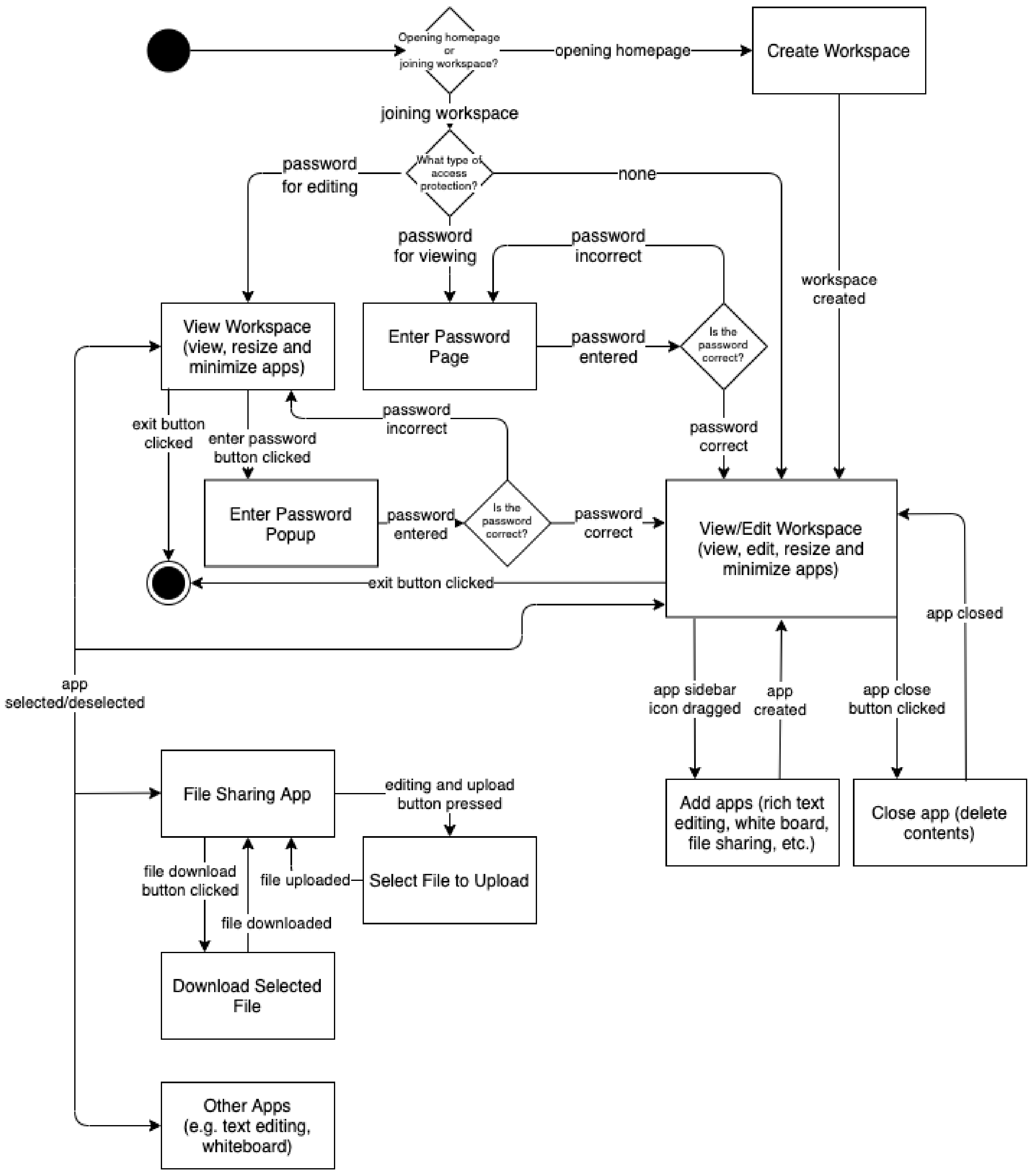


File Upload Application Sequence Diagram

Using the file upload application:

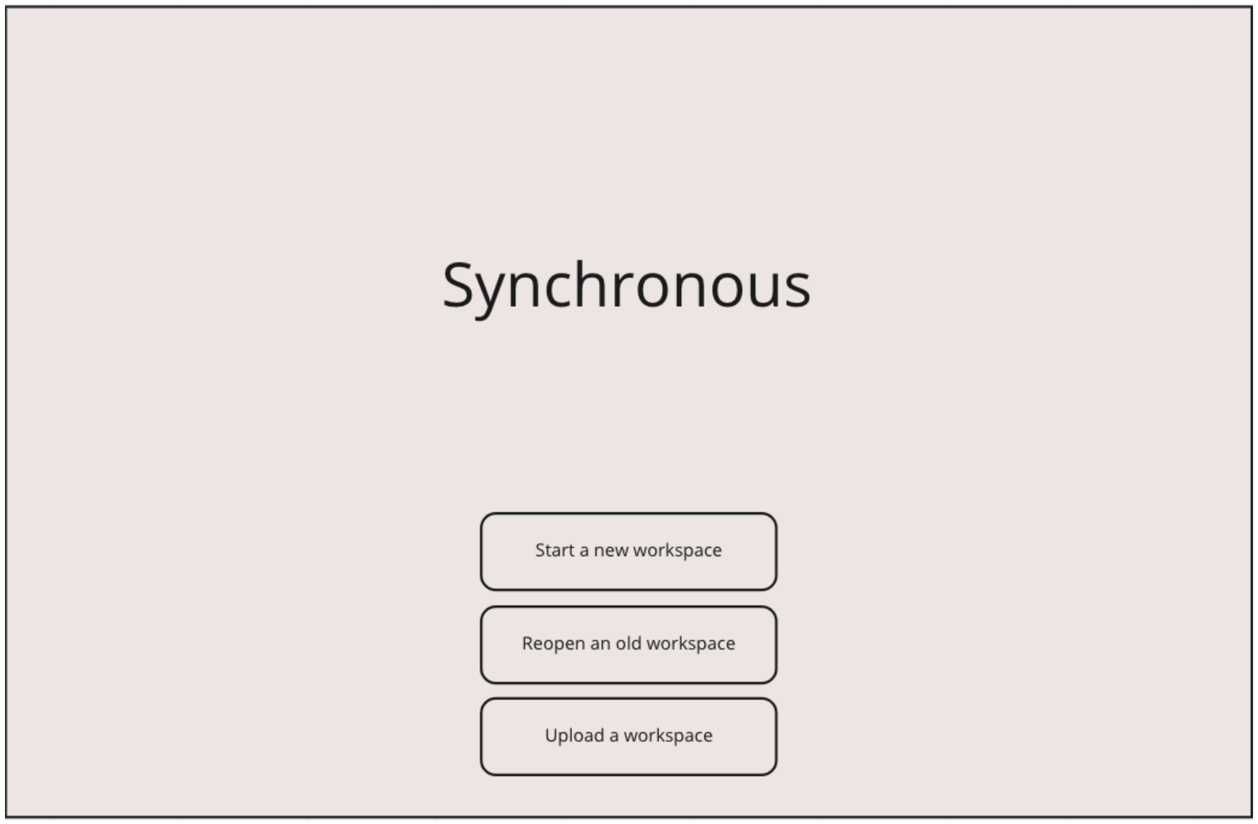


Activity Diagram

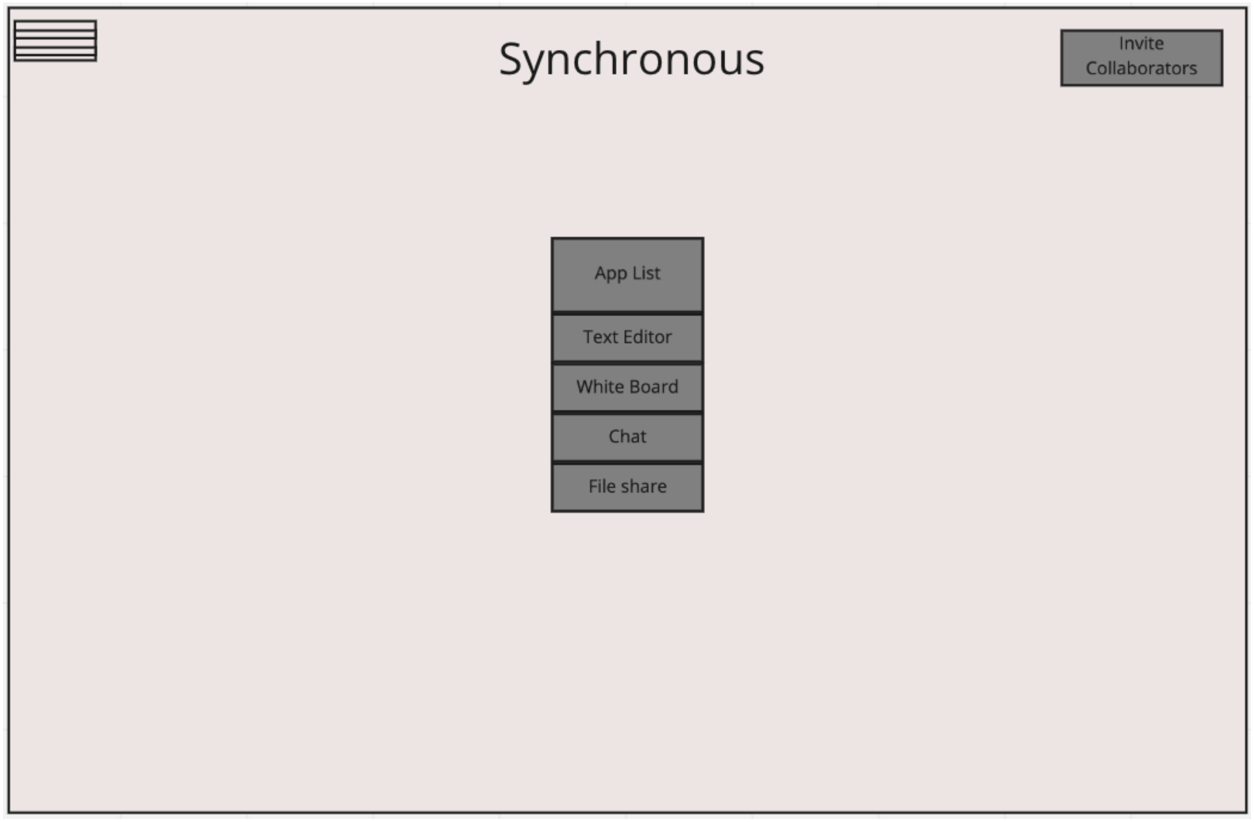


UI Mockups

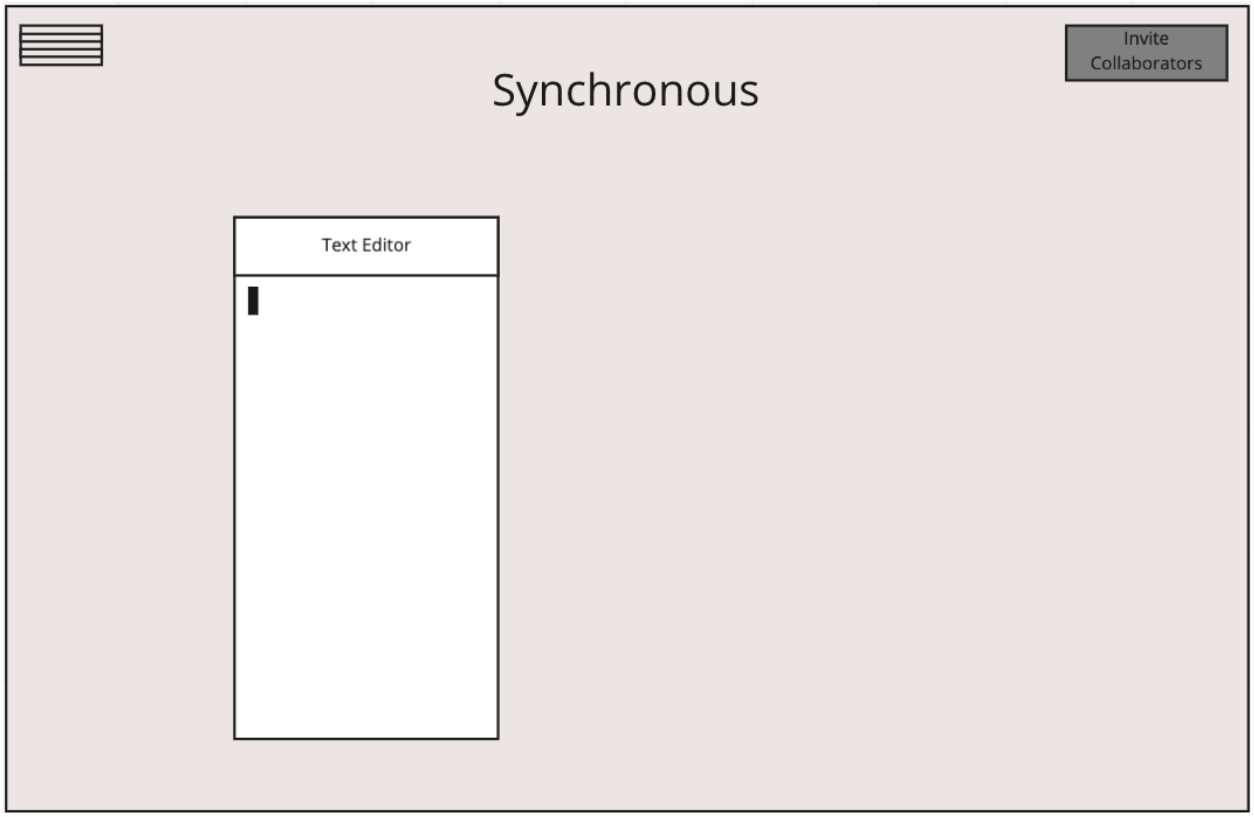
Welcome screen:



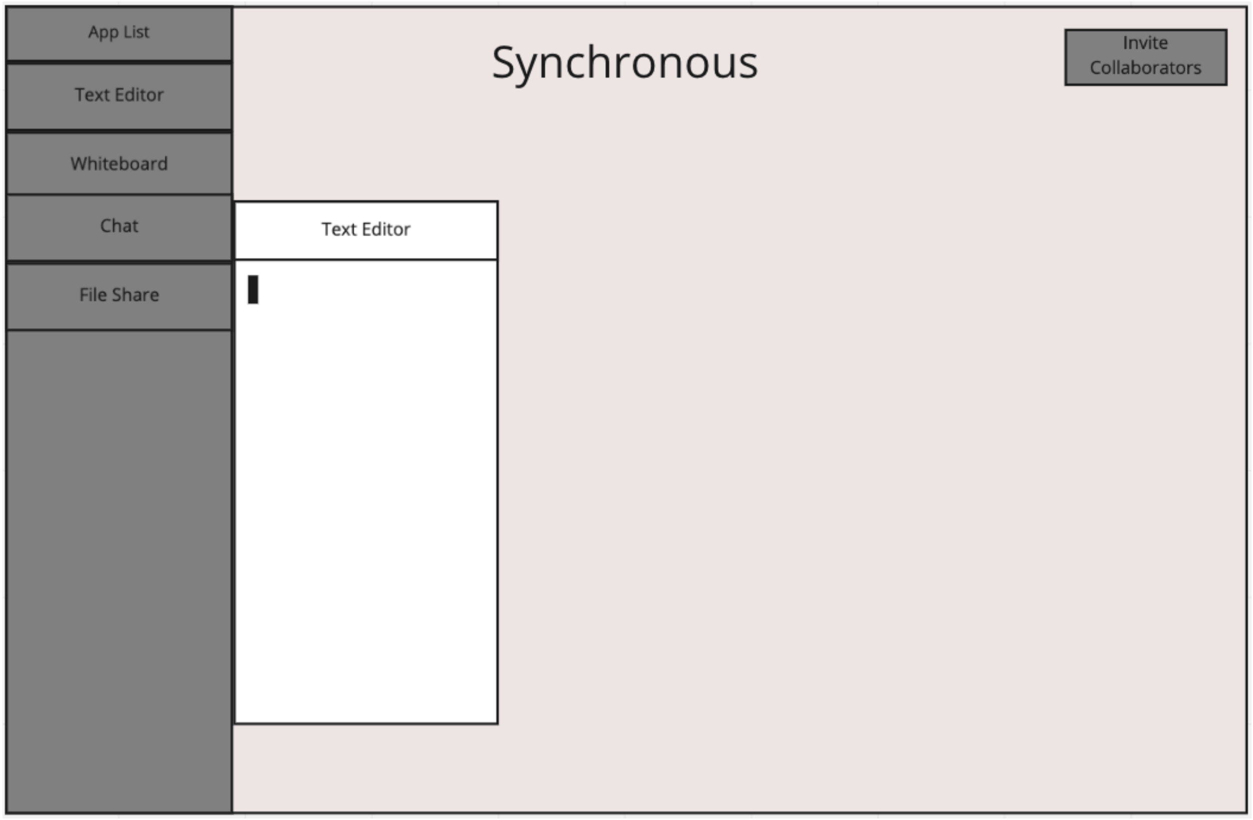
After clicking on “Start a new workspace”:



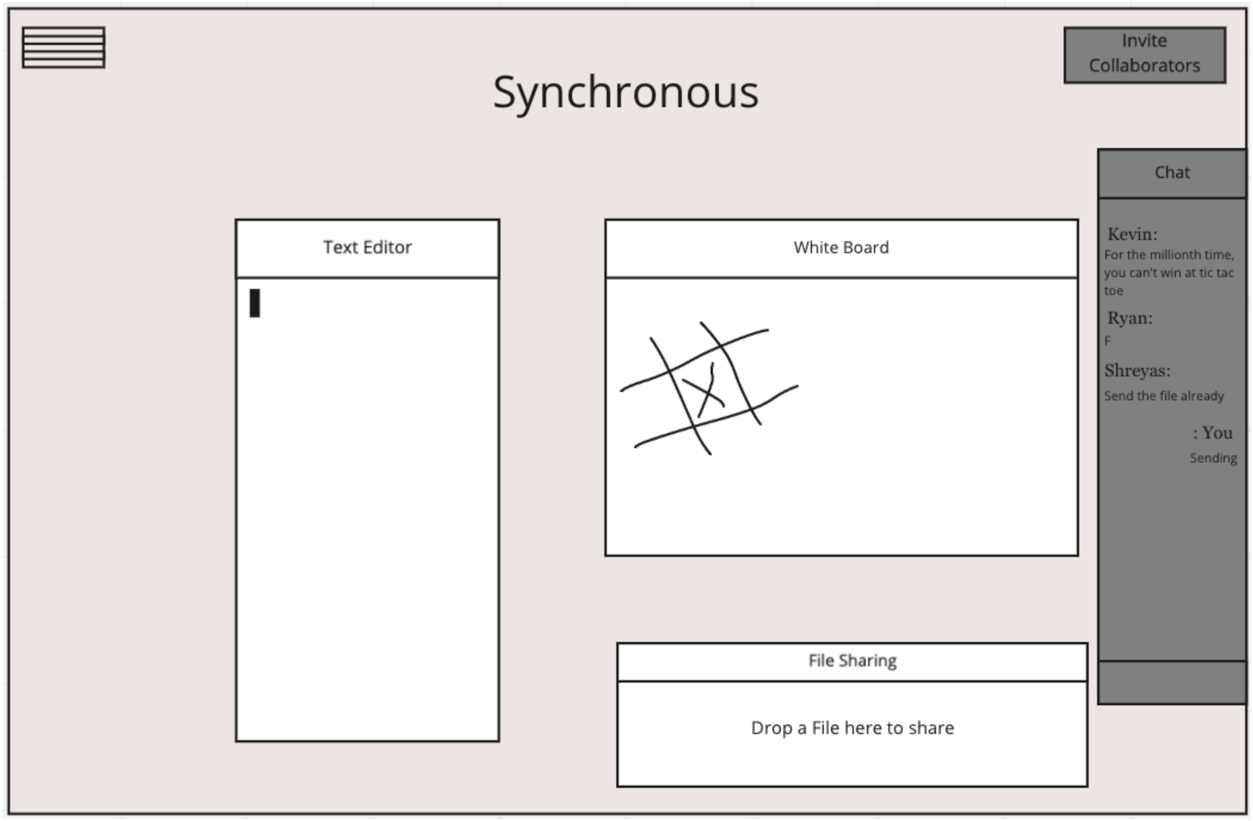
After clicking on “Text Editor” from the app list. The app list also minimizes to the menu icon on top left:



After clicking on the menu icon on the top left:



All apps on screen. You can have multiple instances of an app at the same time:



Kevin:
For the millionth time, you can't win at tic tac toe

Ryan:
F

Shreyas:
Send the file already

: You
Sending