Logan Mann, Naif Alassaf
CSCI 4448
4/13/2022

Project 6 Update

Project Title: CloudCanvas

Team Members: Logan Mann, Naif Alassaf

Work Done:

Thus far, we have almost entirely completed the main backend service/REST API which handles account creation/authentication, canvas creation, adding shapes to canvases, sending updates on canvas state to the client application, etc. We will continue to add functionality to the API as we continue with development as we find areas that need to be fleshed out (adding users to private canvases, etc.) but at present, the backend service has most of its core functionality implemented. We have created a PostgreSQL database to hold user information, and have successfully utilized Spring Data JPA to facilitate access to this database. We have also implemented the Observer and Singleton patterns in our backend to facilitate clean logging of key events in the backend service, with different message severities (INFO, DEBUG, ERROR, etc.) and dates-times for each message. Our logging system outputs these logs to log files by day. In addition to the core functionality of the backend API, we have fleshed out the data model classes which are used to track state within the application, such as Canvas objects, various Shape class implementations, User objects, etc.

Work By Teammember:

Logan Mann: Backend API endpoints and business logic (Create Canvas, Add Shape, Get Canvas, etc.), Auth API (Login and Create Account), creating user database, Observer pattern and Logging, creation of Event classes as part of Observer pattern.

Naif Alassaf: Creation of various model classes (User, Shape implementations, Canvas, etc.), initial implementation of HomescreenController class for when we commence work on front-end.

Both Team Members: Research on SpringBoot, Spring JPA, Dependency Injection, etc. for use throughout the backend SpringBoot application

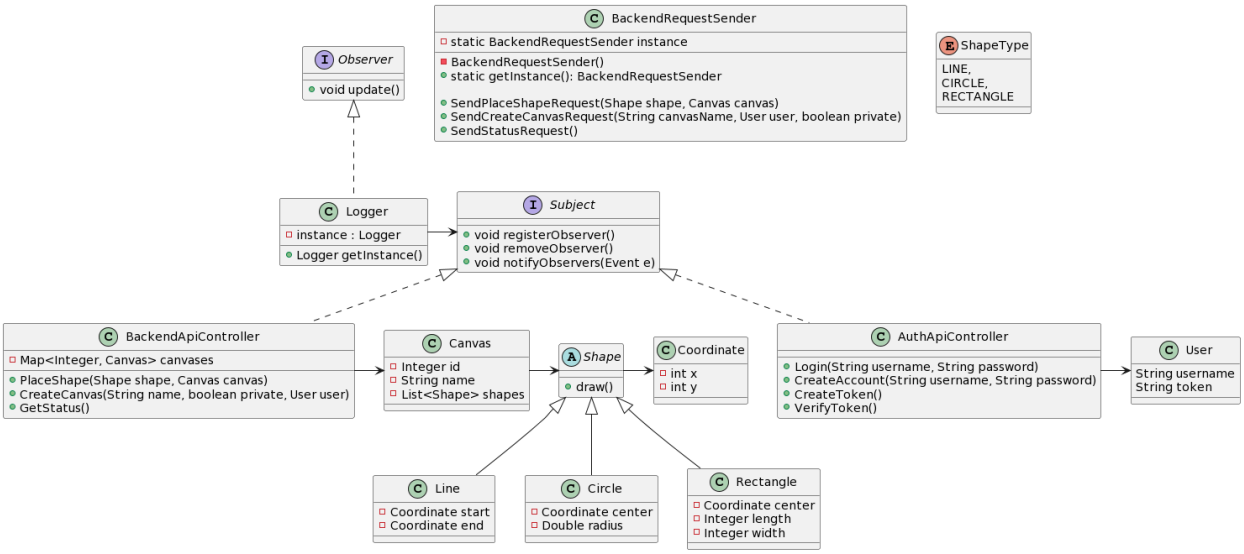
Changes or Issues Encountered:

At present, we are closely following our initial design with no major changes. This was not an issue, but we did have to do a fair amount of research into Spring as part of development of our application backend. We found this to be quite interesting and a good learning experience!

Patterns:

In our work so far, we have implemented the Observer pattern as part of our backend logging system, and the Singleton pattern as part of both Logging and the API Request Sender objects which will send requests from the client to the backend. Our various RestControllers which handle API requests add a singleton Logger object as an observer, and publish events which are then saved to log files by the Logger. When we commence work on the front-end client application, we will be implementing the Command and MVC patterns to process actions from the user, and handle updates to the GUI, respectively. These design patterns are helping us keep our service extensible and maintainable, and keep cohesion tight in the various components of our system. Thus far, the Observer pattern has been extremely useful for keeping logging easy, and separate from the business logic of our API.

Class Diagram:



Plan for Next Iteration:

In order to have implemented the design we presented in Project 5, the majority of the work we have left is in implementing the GUI and the client-application. We will have to implement the MVC pattern/classes for our Auth and Homescreen views, as well as implement the logic to draw the shapes present on a particular canvas on the user's screen. We will also have to implement the Command pattern/logic for responding to user input and publishing these updates to the backend API to update the state of the system. By 4/27, we plan to have a working client application implemented that allows a user to create an account, login, create a canvas, and add shapes to said canvas, all of which will be persisted in our backend service, allowing multiple clients at once to work on the same canvas.