**Project 4.1 README**

**Marco Pagani**

**Nicholas Kroeger**

How to run tests: run these two commands:

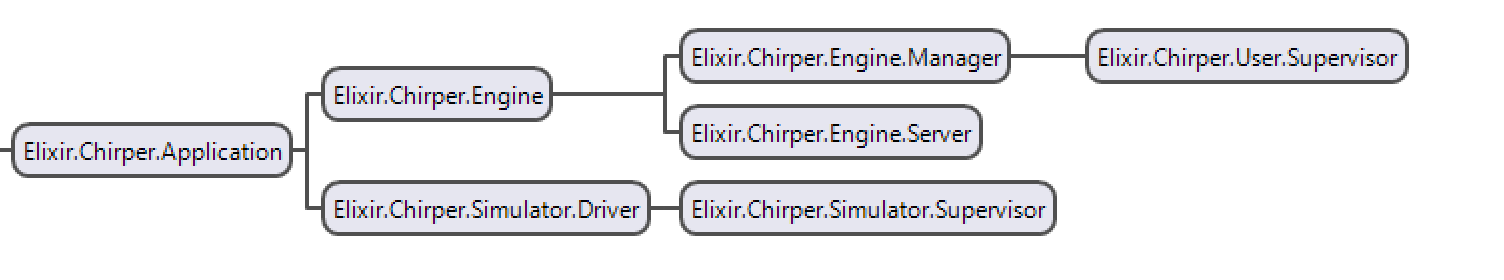
1. Change directory to the top level of the project tree
2. Run “**mix test**”

How to run the driver/simulator

1. Change directory to the top level of the project tree
2. Run “**mix run Chirper.exs num\_user num\_tweets**”

Summary of our implementation

We have all functionality working at this time, including both bonus features.

The structure of the program is as follows

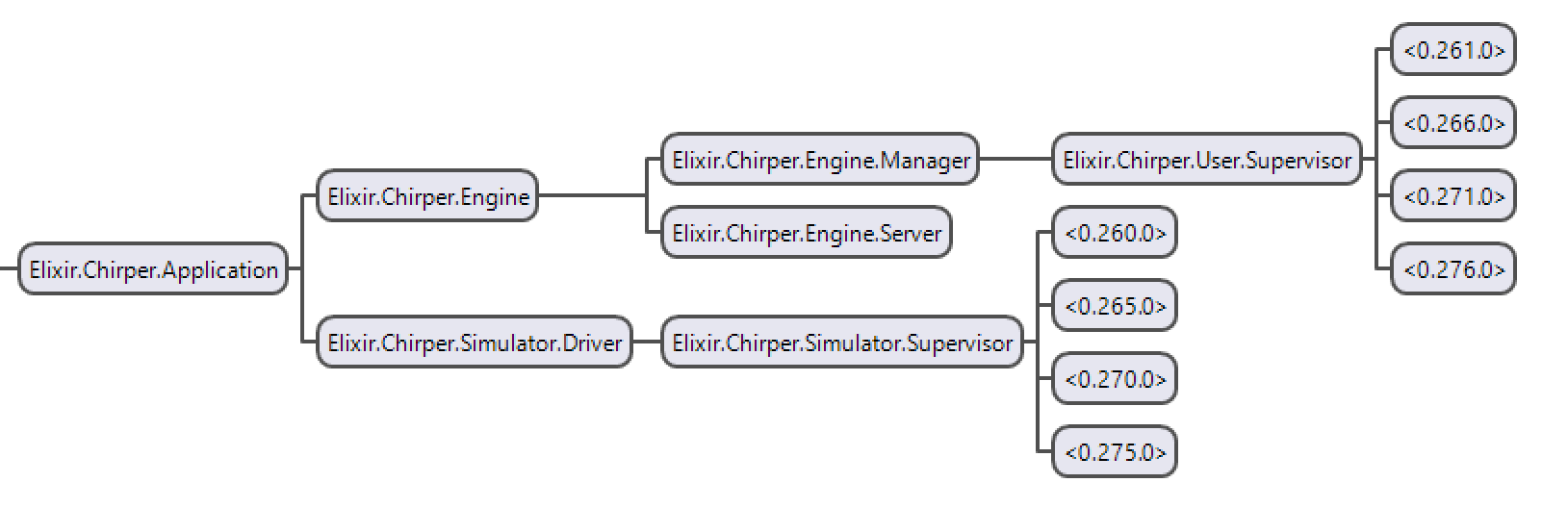
Application – this is an elixir application that will automatically spawn the structure seen above when it is launched. It is launched automatically when iex is opened with the -S mix flag.

Engine –

The engine is the actual twitter service. The server process writes data to a set of :ets tables that store data about tweets and users, as well as their interactions. All requests for reading/writing of data pass through the server.

The manager oversees user registration/deletion. It spawns a dynamic supervisor, and each registered user has its own process managed by this supervisor. This user process maintains state for that user and serves as a client for the user to make requests to the server.

Simulator – The simulator is composed of a driver which controls a supervisor which oversees the simulated user processes (called Actors in our implementation). The driver generates all the data for each of the users in the simulation and instructs the supervisor to spawn the processes for these users before populating them with the data they need to run. The simulation portion of the code is designed so it can easily by substituted for websockets in the coming project without having to modify the engine.

Below is an example of the program structure when 4 users are currently active

The User Supervisor manages the serverside clients that perform operations at the request of the “actors”, which are managed by the Simulator Supervisor.

All required features are implemented. Each has the necessary tests to demonstrate their functionality, specifically:

* Creating accounts
* Deleting accounts
* Sending and parsing tweets
* Following a user
* Distributing messages in realtime to followers
* Using hashtags
* Querying for hashtags
* Mentioning a user
* Retweeting from dashboard

Bonus features working and tested:

* Maintaining state when user disconnects and rejoins
* Generating zipf distributions
* Simulating user activity based on zipf distributions