Message In A Bottle

Team Members

- Geng Sng (gs148)
- Gerald Kora Kwok (gk72)

Important Links

- Repository: https://github.com/duke-compsci290-spring2018/final-project-team-36
- Gh-pages: https://duke-compsci290-spring2018.github.io/final-project-team-36/

Brief Description

We would like to build an online message-in-a-bottle platform similar to https://paperplanes.world. Users will be able to place short messages inside virtual bottles, retrieve a bottle at random, read the message contained inside, and add their own message.

This is interesting because it is a platform that can connect people all over the world. Ideally we'd like to implement some cool visualizations to make the platform look aesthetic (explore tools like three.js). Our data sources will be the user-inputted messages, but other ways we could incorporate existing data sources could be scraping text from popular sites (reddit, etc) to fill initial bottles, or making the 'ocean' environment dependent on real-time data sources like weather.

Technology Stack

- Backend
  - Node.js
  - Express.js
  - MongoDB
  - Mongoose.js
  - Passport.js
- Frontend
  - HTML, CSS, Vanilla Javascript
  - Semantic UI
  - (Maybe) React.js or Vue.js
- External APIs
  - Weather -- AccuWeather

User types

- ***Guest***
  - View bottles
- ***User***

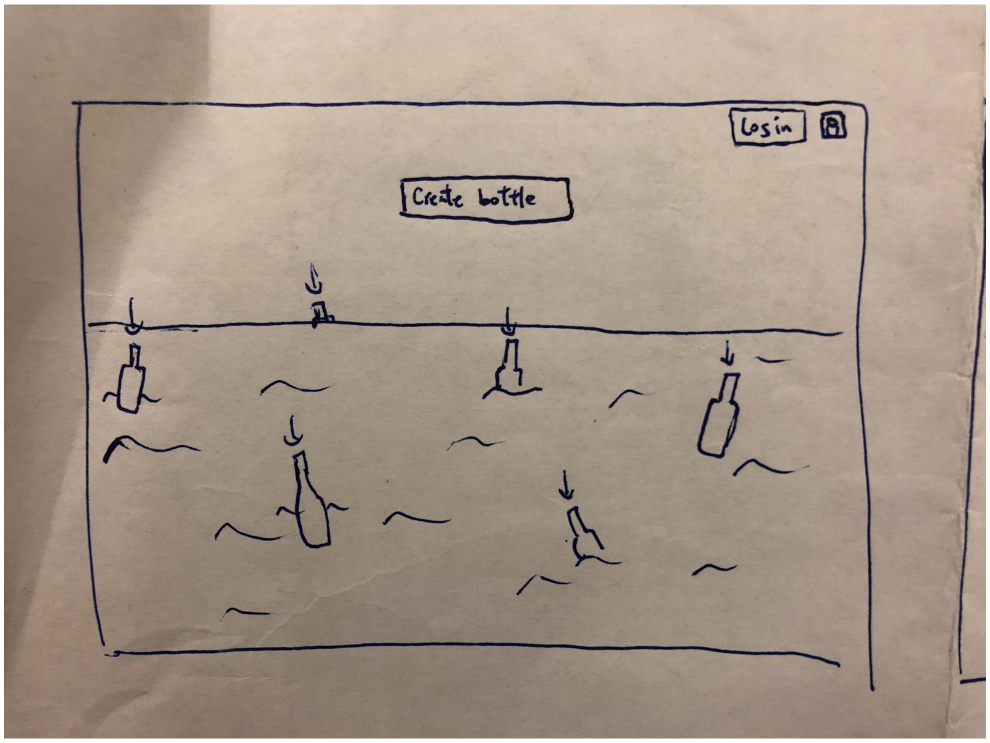
    - View bottles
    - Edit existing bottles (add to bottle message)
    - Create new bottles
    - Potential preferences:
- ***Administrator***
    - All the above
    - Delete bottles, edit bottle messages
    - Delete users, edit user preferences

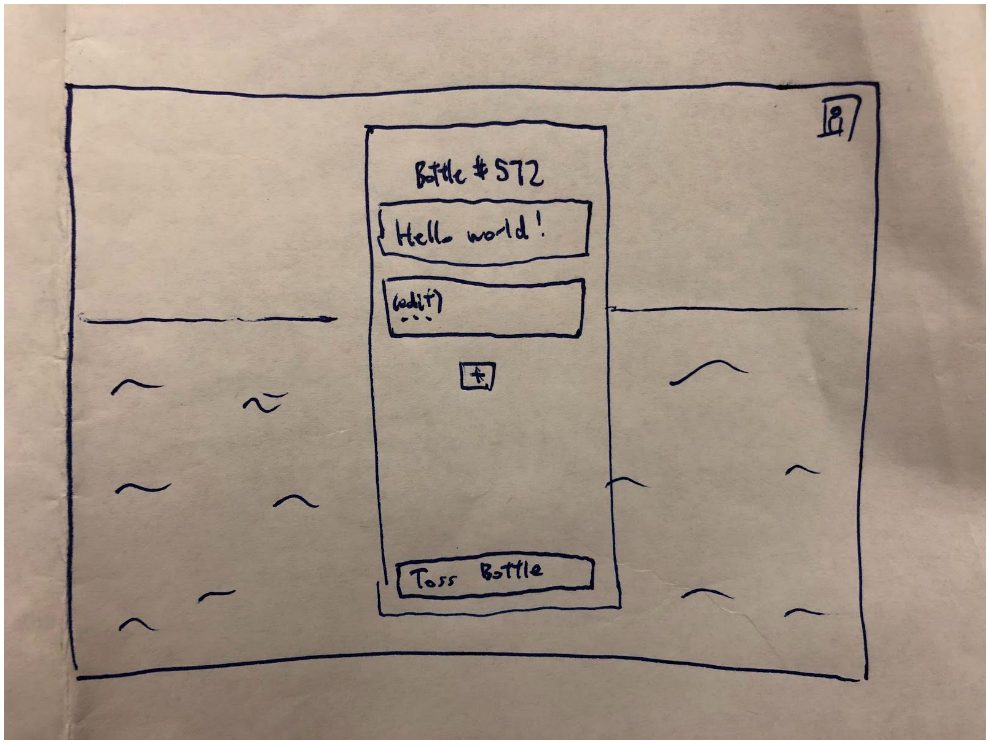
Features

- User Login
    - Framework(s): Node.js, Express.js, MongoDB, Mongoose.js, Passport.js, Bcrypt.js, JWT
    - Permissions: User, Admin
    - User login will be enabled via a REST API which connects to our MongoDB backend
    - Mongoose.js will be used as an ORM
    - Our database will have a UserSchema which creates a UserModel, with its own controller which is accessible via a route exposed by Express.js, following the MVC convention
    - User creation will be salted and hashed using bcrypt.js and passport.js
    - Authorization middleware will be implemented to check for user roles and persisted onto the frontend using JWT and local storage
    - User object is exposed on successful login within local storage, which allows us to identify user permissions
- Display 3D Environment, Bottles
    - Framework(s): Three.js, potentially A-Frame
    - Permissions: Viewer, User, Admin
    - Create stacked or layered environment that simulates waves / ocean using three.js which uses WebGL to create 3D Graphics
    - Bottles will be constructed javascript objects with ids that map to respective bottles within our database, such that individual bottles are identifiable when users click on them
        - If the mapping proves too complicated, we may skip the object-bottle mapping and just retrieve a random bottle whenever a user clicks on any bottle
    - Note: we may end up using A-Frame to display the virtual environment; will have to do a bit more exploring first to decide if this framework is suitable
- Display Message View
    - Framework(s): Three.js, Semantic UI
    - Permissions: Viewer, User, Admin
    - Clicking on bottles will trigger a modal to show the detailed information about that bottle

    - Either create animation using Three.js, or just use Semantic UI modal
- Creating Bottles
    - Framework(s): Node.js, Express.js, MongoDB, Mongoose.js
    - Permissions: Users, Admins
    - Bottle creation will be enabled via a REST API which connects to our MongoDB backend
    - Mongoose.js will be used as an ORM
    - Users can create bottles with custom messages
    - Our database will have a BottleSchema which creates a BottleModel, with its own controller which is accessible via a route exposed by Express.js, following the MVC convention
- View User Data and Bottle Data
    - Framework(s): Node.js, Express.js, MongoDB, Mongoose.js
    - Permissions: Admins
    - Admins can view all users and all bottles
    - There will be a getAll function in both controllers for users and bottles to do this
    - Display number of users and number of bottles
- Users can update preferences
    - Framework(s): Node.js, Express.js, MongoDB, Mongoose.js
    - Permissions: Admins, Users
    - Users will be able to update preferences like a background color, or “type” of bottle
    - There will be an updatePreferences function in the user controller to do this
- Changing Weather
    - Framework(s): Weather API, Three.js, Semantic UI
    - Permissions: Viewers, Users, Admins
    - Our application’s ocean environment will display weather changes based on the current weather
    - It will show the date, weather, and temperature in a sidebar
    - The “weather” inside our environment will be affected by this dynamic weather data, and is rendered dynamically using Three.js

Quick mockups





------- proposal end -------