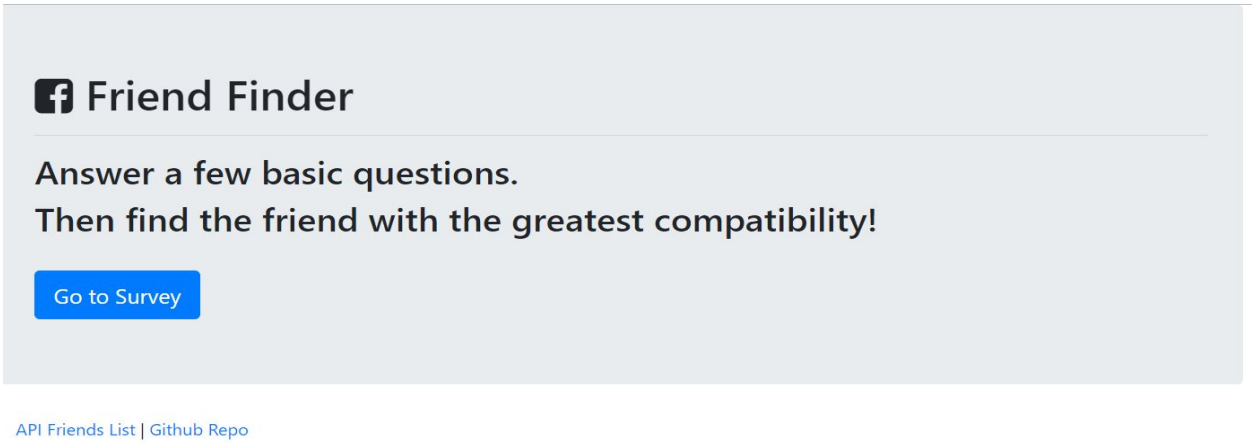
Friend Finder - Node and Express Servers

In need of a friend? This is the app for you. Find that special someone. After filling out a quick survey, our friend-matching algorithm will pair you with an individual in our network.

Live Link

https:#.herokuapp.com/

Usage



To use our web service, simply go to our homepage and take our state-of-the-art survey. Immediately after submitting the survey, your perfect best friend will pop up. We also have an API you can access to the network's users and their personalized information. For research purposes.

Requirements Modularity in the form of separate files for server logic, storing of friends, views, and routing 10-question survey to assess uniqueness of users Use express,, and path npm packages in the server.js file Separate JavaScript files for routing (htmlRoutes.js and apiRoutes.js) Appropriate GET and POST routes for serving HTML pages and API callshttps:#herokuapp.com/ Separate file for storing friends (friends.js) Calculate best match for user once survey is completed and return that match to the user Technologies Used JavaScript jQuery node.js Express.js HTML Bootstrap Code Explanation Our server.js file sets up the Express server, specifying our port number, the npm packages that need to be loaded, and also the routes, which we have externalized There are 2 separate HTML files (home.html and survey.html) that serve

as the front-end portion of our code; they determine what the user sees (the homepage and the survey, which will also show the resulting best match) Our 2 routing files (htmlRoutes.js and apiRoutes.js) determine the back-end logic (based on the request being made, the response that gets sent to the browser); the HTML routes display the survey and the homepage based on the URL that is accessed, and the API routes send back existing content in our server-side data or add new friends Best match is calculated by finding the friend with the minimal difference in scores and then sending that friend to the browser as a JSON object A modal is then toggled, displaying the the best match to the person who just took the survey In essense, this will also be a form of notes that you may later reference weeks later Friends are stored as such:


```
{"name": "Mike Smith",
"photo":"https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjo1syV-9LfAhUl4oMKHQchBoAQjRx6BAgBEAU&url=%2Furl%3Fsa%3Di%26source%3Dimages%26cd%3D%26ved%3D%26url%3Dhttps%253A%252F%252Fopenclipart.org%252Fdetail%252F261880%252Fcartoon-man-avatar%26psig%3DAOvVaw14MrPT3x7zqFfZhdP9y2OL%26ust%3D1546188099977343&psig=AOvVaw14MrPT3x7zqFfZhdP9y2OL&ust=1546188099977343",
"scores": [5,5,5,5,5,5,5,5,5,5]},
```