I’ve spent too much time trying to make this as simple as possible to operate without spending another couple of weeks making a user interface like the one I designed. I’ve included steps in the project folder that first builds the project, then runs the API (which also initializes a LocalDb), and then runs a console application that interfaces with the API to create a user profile.

1. Go into main folder
2. Click Step1.bat
   1. It will quickly build the bin folder and then exit
3. Click Step2.bat
   1. It will start the ResumeBuilder project
      1. This includes the user interface that I started building from an existing template, but focused more on making the API fully functional
      2. If you really wanted to, you could open up a browser and navigate to localhost:5000, but you’re really only getting a template
      3. The API runs on localhost:61171/api
         1. If you’re fluent with REST calls, you could use the routes in each of the Controllers in the “Controllers” folder
      4. Wait until the console says that it has “Compiled successfully”
4. Click Step3.bat
   1. This will start an interactive console window that will ask for your information if you’re unfamiliar with REST calls
   2. It will return the object that is stored in the LocalDb instance populated with the information you entered

STATUS

* The database and the API could be considered deliverable. The user interface is not deliverable at all.
* Problems encountered were many. I didn’t know about a lot of the technology that went into this. The learning accounted for 90% of my time in this project. I couldn’t figure out how to set up a locally running database and interface with it with the API. The API had its own problems with trying to use .NET Core 2.2 for the first time and using code-first migrations to establish that locally running database.
* I think the remaining stages, in priority order, are:
  + Create the user interface
  + Catch all exceptions and add logging and tests
  + Find out how to deploy the parts separately for easier managment