U0JBSEGHY : I feel like I have gotten really close. I'll show the error that was tripping me up.
U3SJEDR96 : `field "token" string` is all you need
U3SJEDR96 : which decodes that JSON into simply the string `XX_SECRET_TOKEN_XX`
U0JBSEGHY : ```fetchToken =   Http.send ReceiveToken (Http.get apiEndpoint tokenDecoder)

tokenDecoder =
  Decode.decodeString (Decode.field "token" Decode.string)```

U0JBSEGHY : Says function `get` is expecting `Decode.Decode a` but is getting `String -> Result String String`
U180KMGRE : You don't want the decodeString there
U180KMGRE : decodeString is partially applied so it then wants a string to decode, but Http.get will run the decoder when it gets the response and so on, so it only needs the decoder to run
U0JBSEGHY : sorry I am not sure I follow
U180KMGRE : decodeString is trying to apply the decoder to a string - you don't need that to happen - you just want the decoder `Decode.field "token" Decode.string`
U0JBSEGHY : Was following here and various other examples. Which had variants of the solution.
<http://package.elm-lang.org/packages/elm-lang/http/latest>
U0JBSEGHY : ```The 2nd argument to function `send` is causing a mismatch.
86| Http.send ReceiveToken (Http.get apiEndpoint tokenDecoder)
                  ^^
Function `send` is expecting the 2nd argument to be:

  Http.Request Model

But it is:

  Http.Request String```

U153UK3FA : <@U0JBSEGHY> ReceiveToken takes a Model not a String
U0JBSEGHY : Yea this is where I keep tripping up in all the various ways I have tried
U153UK3FA : You need to change ReceiveToken so it takes a String
U0JBSEGHY : My model contains other stuff... which isn't in the Json response
U180KMGRE : Can ReceiveToken take a string and then in your update you can update it in your model?
U0JBSEGHY : OH!!
U0JBSEGHY : Damn Types.... haha
U0JBSEGHY : still getting used to it :sweat_smile:
U153UK3FA : Look at the type of Http.send
U0JBSEGHY : I had this `| ReceiveToken (Result Http.Error String)` saying Model. Which I have now changed.
U0JBSEGHY : Ok I'm getting an error / not getting anything back when I make the request :disappointed:
U3SJEDR96 : Try logging the `Err` case to the console using `Debug.log`, or inspecting the messages in the debugger
U0JBSEGHY : Where would that fit into code like this? ```type Msg   = Answer
  | HangUp
  | FetchToken
  | ReceiveToken (Result Http.Error String)
  | Err
  | NoOp

update : Msg -> Model -> (Model, Cmd Msg)
update msg model =
  case msg of
    Answer ->
      ({ model | log = Connected }, Cmd.none)

    HangUp ->
      ({ model | log = Disconnected }, Cmd.none)

    FetchToken ->
      (model

```
        , fetchToken
        )

    ReceiveToken (Ok newToken) -&gt;
        ( { model | token = newToken }
        , Cmd.none
        )

    ReceiveToken (Err _) -&gt;
        ( model
        , Cmd.none
        )

    Err errorMessage -&gt;
        let
            _ =
                Debug.log "Error in HTTP response: " errorMessage
        in
            NoOp

    NoOp -&gt;
        ( model, Cmd.none )


fetchToken =
    Http.send ReceiveToken (Http.get apiEndpoint tokenDecoder)


tokenDecoder =
    Decode.field "token" Decode.string```

U0JBSEGHY : yea I ended up with something similar.  ```ReceiveToken (Err errorMessage) -&gt;  let
            _ =
                Debug.log "Error " errorMessage
        in
            ( model
            , Cmd.none
            )```
Says that `Err` has too many arguments.

U0JBSEGHY : Ok think I got it compiling now
U3FJSB596 : Can you paste the log message?
U0JBSEGHY : My beginner web programming experience is showing xD
U0JBSEGHY : `XMLHttpRequest cannot load <http://localhost:4000/api/twilio/token>. No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin '<http://localhost:3000>' is therefore not allowed access.`
U3SJEDR96 : Ahh, good old CORS.
U0JBSEGHY : On MDN now :laughing:
U3SJEDR96 : you'll have to configure your server to allow ajax requests from `localhost:3000`
U0JBSEGHY : dope
```