

Introduction

The primary objective of this project is to implement a simplified version of Bond Street's bank login flow. The final product should look identical to the screenshots included. Inside the zipfile is a nodejs server that implements the API endpoints you'll be connecting to retrieve login fields and post login credentials.

Server Setup

1. Install Node and NPM if you haven't already (instructions can be found here: <http://blog.npmjs.org/post/85484771375/how-to-install-npm>)
2. Navigate to the bank_login_server/ folder and run the following commands:

```
npm install  
npm start
```

You can navigate to the server by visiting <http://localhost:3000>

Flow Steps

After you've got the server running, your task is to implement the following steps with HTML, CSS and javascript.

Step 1: Landing Page

When the customer initially loads the page, they should see the following page:
(all screenshots are also present in the screenshots directory)

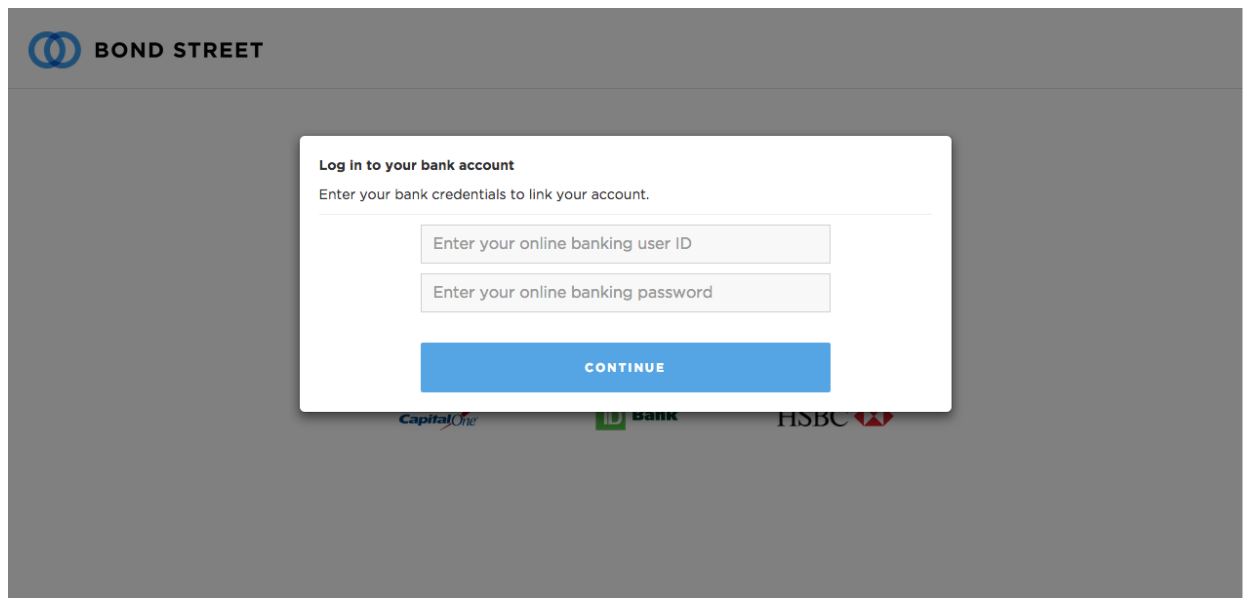
Connect Bond Street to your bank

Connecting us to your primary business bank account gives us a clearer understanding of your finances.



Step 2: Render the Login to Your Bank Account Form with retrieved JSON

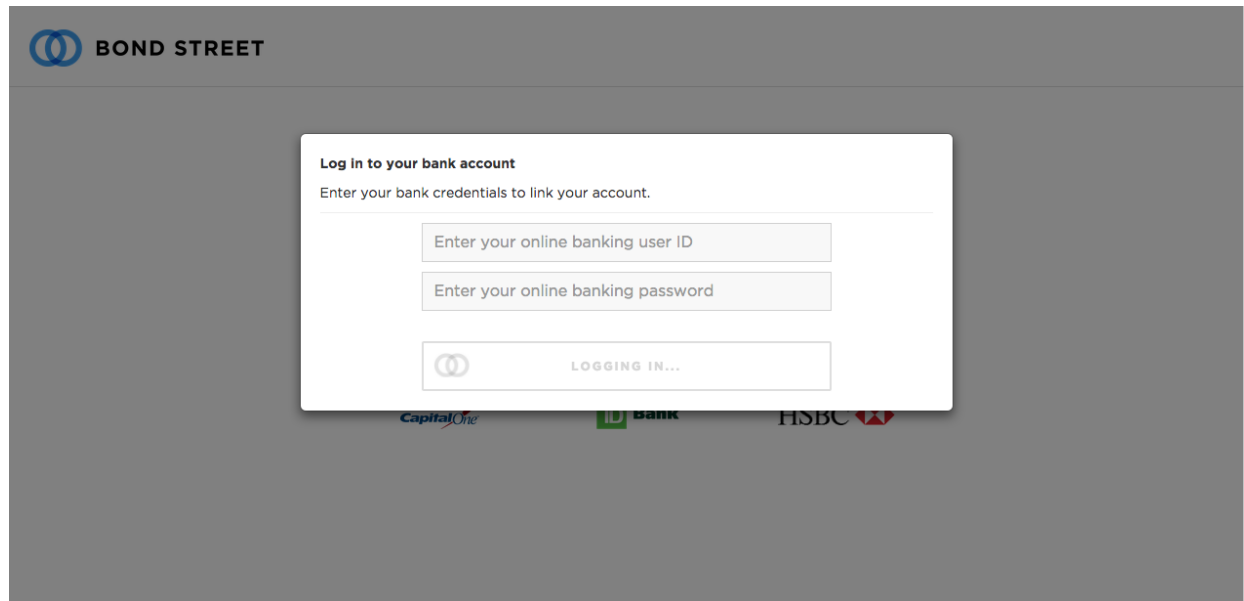
When one of the bank icons is clicked (located in screenshots/icons/bank_icons), the client page should call the `/bank_login_fields` endpoint to retrieve the fields level details of the form in JSON. This should be used to render the following:



The screenshot shows a web interface for Bond Street. At the top left is the Bond Street logo. The main content area is a light gray background. In the center, there is a white modal box with a blue border. The modal box has the title "Log in to your bank account" and a subtitle "Enter your bank credentials to link your account." Below the subtitle, there are two input fields: "Enter your online banking user ID" and "Enter your online banking password". At the bottom of the modal box is a blue button labeled "CONTINUE". Below the modal box, the logos for Capital One, TD Bank, and HSBC are visible.

Step 3: Form Submission Loading State

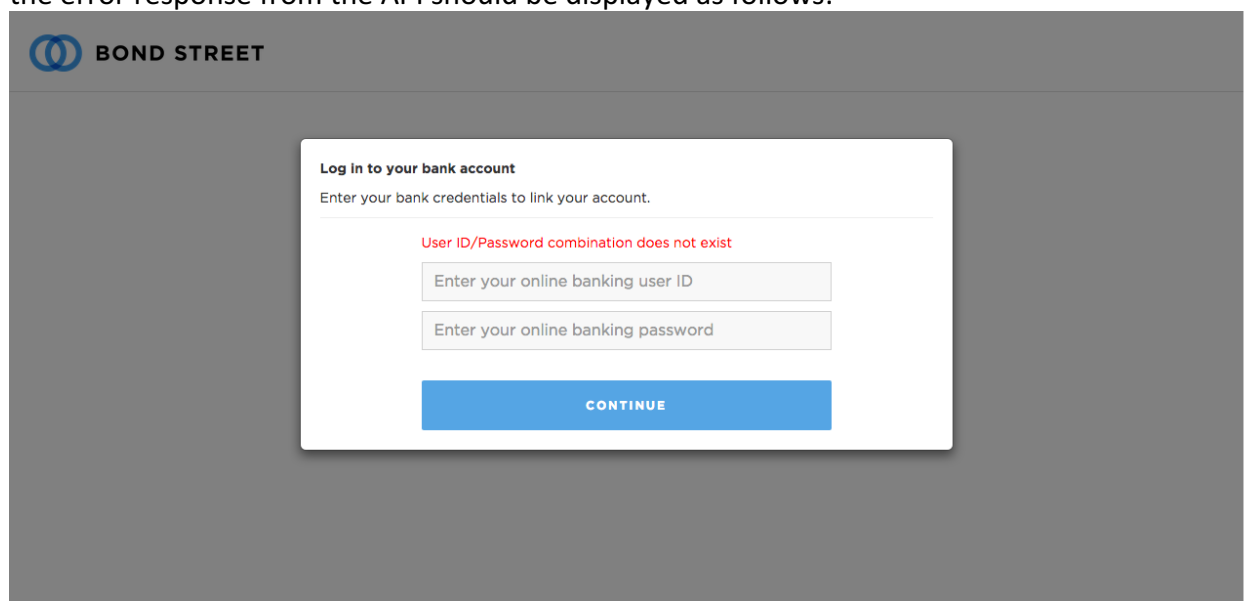
After the user submits a username and password, show the loading state below (loading GIF provided in the screenshots folder). Submitting this form should send a post request to the /bank_login_creds endpoint (see Server API Documentation PDF for more details)



The screenshot shows a web interface for Bond Street. At the top left is the Bond Street logo. Below it is a white modal box titled "Log in to your bank account" with the instruction "Enter your bank credentials to link your account." Inside the modal, there are three input fields: "Enter your online banking user ID", "Enter your online banking password", and a third field containing a loading spinner and the text "LOGGING IN...". Below the modal, the logos for Capital One, ID Bank, and HSBC are visible.

Step 4: Form Error State

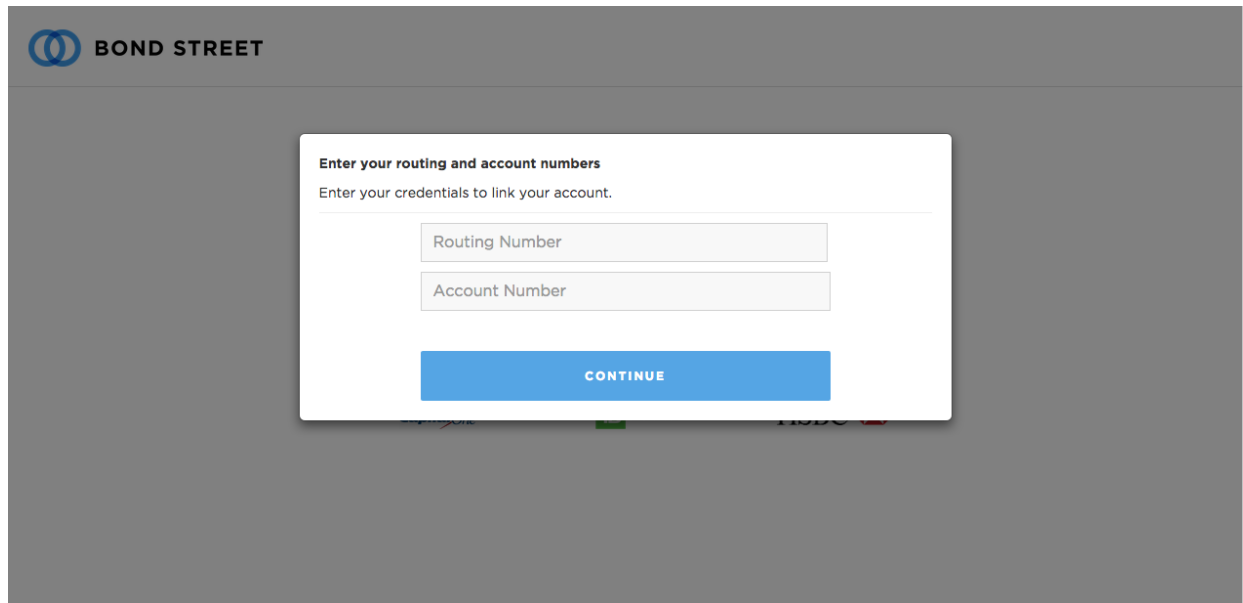
If the user enters anything besides "bondstreet" as the the user_id (password can be anything), the error response from the API should be displayed as follows:



The screenshot shows the same Bond Street login interface as in Step 3. The modal box is titled "Log in to your bank account" with the instruction "Enter your bank credentials to link your account." Below the instruction, there is a red error message: "User ID/Password combination does not exist". There are two input fields: "Enter your online banking user ID" and "Enter your online banking password". At the bottom of the modal is a blue button labeled "CONTINUE".

Step 5: Bank Routing Information

Following a successful bank login, the modal should transition to the account details page where the customer can supply their routing and account numbers.



The screenshot shows the Bond Street app interface. At the top left is the Bond Street logo, consisting of a blue circular icon with two interlocking loops and the text "BOND STREET" in a bold, sans-serif font. The background is a solid dark gray. Centered on the screen is a white modal box with a subtle drop shadow. Inside the modal, the text "Enter your routing and account numbers" is displayed in a bold, dark gray font. Below this, a smaller line of text reads "Enter your credentials to link your account." The modal contains two input fields: the first is labeled "Routing Number" and the second is labeled "Account Number", both in a light gray font. At the bottom of the modal is a solid blue button with the word "CONTINUE" in white, uppercase, sans-serif font.

Step 6: Success Page

Once the account details are submitted, the application should close the modal and show the final success view.

Connect Bond Street to your bank



You've successfully connected your Bank

When you are approved for your loan, Bond Street will use your bank for payments. You can change this later once you are approved.

BACK

CONTINUE

Extra Credit (optional)

- Validate the routing number using the formula below via client side validation. d_n stands for digit in position n . Show an error message if the routing number is not valid – 011111111 is a valid routing number you can test with.

Enter your routing and account numbers

Enter your credentials to link your account.

111111111

Please check your routing number.

Account Number

CONTINUE

$$3(d_1 + d_4 + d_7) + 7(d_2 + d_5 + d_8) + (d_3 + d_6 + d_9) \mod 10 = 0.$$

- Instead of using the loading GIF in step 3, make the loading symbol with a CSS animation.
- Implement the server side API call for POSTing account details. This may simply return stubbed data (you'll need to edit the NodeJS app).
 - Double extra credit - persist data sent via API in some form of database
- Design a better looking modal