E4C iOS App: Backend Implementation Details

Code

https://github.com/JSalter15/E4CiOSBackend

Clone the repo and run the following commands to run locally:

```
npm install
npm start
```

Server

The server is built with Node and Express. All API endpoints can be found in the file **server.js**. See **apidocs.pdf** for detailed documentation on how to use the API, including the requisite parameters for each requests and what the server returns in response. Essentially, each API call in server.js makes a call to one of two database modules to receive the necessary information to return to the client:

1. **database_pg.js**: handles everything related to the Community aspect of the app (users, projects, project comments).
2. **database_e4c.js**: handles all information directly related to E4C's database -- retrieving their news and webinars.

See below for database implementation details.

Database

E4C manages a MySQL database accessed via phpMyAdmin on the Wordpress framework. They provided me with access to a phpMyAdmin online portal that allowed me to view their staging site's database and everything contained in it (they chose their staging sites database because it is not their live database; they did not want us having the potential to alter their live website -- thank God! This was great for development, as the schemas in the staging db are the same as those in the live db). Their database follows the WordPress framework's database schematic, which you can view at this link:
https://codex.wordpress.org/Database_Description

- News are held in the wp_posts table where column post_type = 'post'
- Webinars are held in the wp_posts table where column post_type = 'wwebinars'

However, we were not in time able to develop a way to give me remote access to the database. Therefore, I was unable to connect to the db in my live Node application. As a workaround, I downloaded the tables I needed to retrieve News and Webinars from the phpMyAdmin online portal. Once I had these four tables

wp_posts, wp_terms, wp_term_relationships, wp_term_taxonomy

in CSV files, I was able to use a library called pgloader to load them into my own PostgreSQL database, which I could then access live from my application.

I chose PostgreSQL because it is built right into Heroku, where the entire app is hosted. So, the first component of the database contains the four tables mentioned above, which allow you to retrieve news and webinars and the target sectors (Featured, Water, Health, Housing, etc.) that they are each tagged with.

The second component of the database handles the Community component of the app. These schemas we were designed by me, not copied over from E4C's database. There are three tables in this component of the db: users, projects, and project_comments. The users table contains all of the columns contained in E4C's wp_users table, with some additions like fav_webinars and user_projects. Exact schemas are pictured below:

Users

Column	Type	Modifiers
id	uuid	not null
user_firstname	character varying(50)	
user_lastname	character varying(50)	
user_login	character varying(60)	
user_pass	character varying(255)	
user_nicename	character varying(50)	
user_email	character varying(100)	
user_url	character varying(100)	
user_registered	timestamp without time zone	not null default now()
user_activation_key	character varying(255)	
user_status	integer	
display_name	character varying(250)	
user_age	smallint	
user_sectors	character varying(11)[]	
fav_articles	integer[]	default '{}'::integer[]
fav_webinars	integer[]	default '{}'::integer[]
user_profstatus	integer	default 0
user_affiliation	integer	default 0
user_expertise	integer	default 0
user_description	text	default 'Tell us a little about yourself!'::text
user_gender	integer	default 2
user_country	integer	default 0
user_projects	uuid[]	default ARRAY[]::uuid[]

Projects

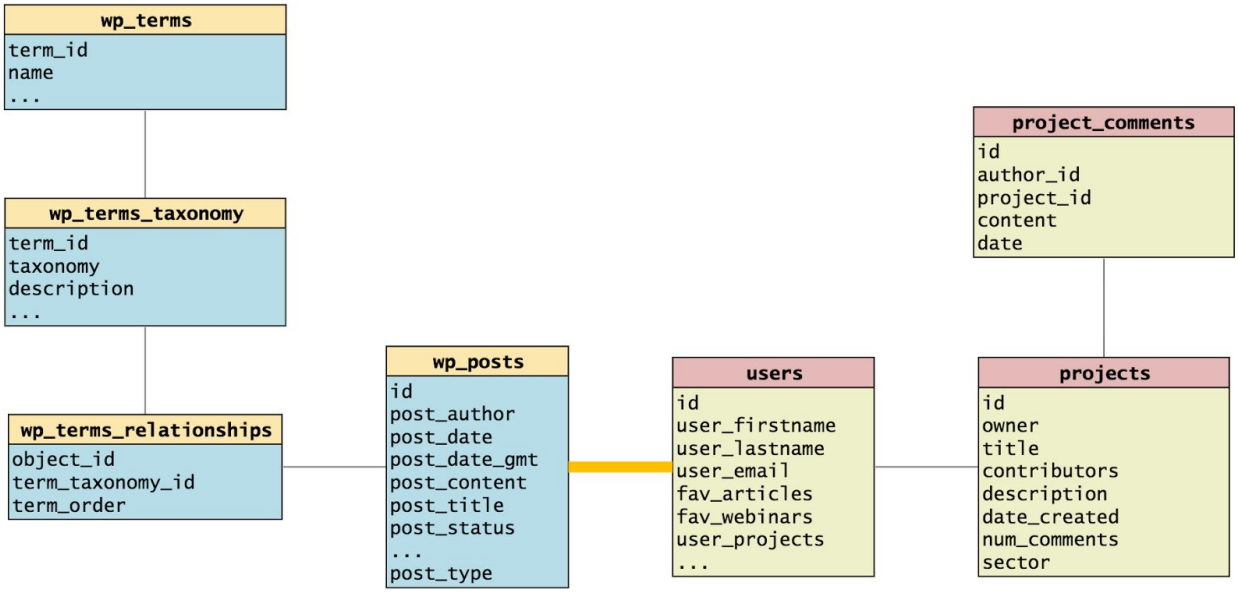
Column	Type	Modifiers
id	uuid	not null
owner	uuid	
title	character varying(200)	
description	text	
date_created	timestamp without time zone	not null default now()
num_comments	integer	default 0
sector	character varying(11)	
contributors	uuid[]	default ARRAY[]::uuid[]
owner_name	character varying(200)	
owner_email	character varying(200)	

Project_comments

Column	Type	Modifiers
id	uuid	not null
author_id	uuid	
project_id	uuid	
content	text	
date	timestamp without time zone	not null default now()

Notice how everything is stored with a uuid, which is Postgres' built in unique identifier data type.

Here is an ER diagram of the entire resulting database:



The blue and yellow tables are for news and webinars; the green and red tables are for the Community. Below explains how all the tables are linked together:

1. 'Object_id' in 'wp_term_relationships' = 'id' in 'wp_posts'
2. 'term_taxonomy_id' in 'wp_term_relationships' = 'taxonomy' in 'wp_term_taxonomy'
3. 'Term_id' in 'wp_term_taxonomy' = 'term_id' in 'wp_terms'

So, a post id gives you its term relationships, which give you its term taxonomies, which give you its terms. Its terms include the sector/s it is tagged with

1. 'fav_articles' and 'fav_webinars' in the 'users' table hold arrays of 'id's' from 'wp_posts'. This is the link between the two components of the database (thick yellow line above)
2. 'user_projects' in the 'users' table contains an array of project 'id's'
3. Comments are linked to projects by 'comment_id' in the 'project_comments' table

The PostgreSQL database is accessed in the Node application using node-postgres.

Limitations and Recommendations

1. Get remote access to E4C's database before you begin working! To do this, you will need to be working from a static IP address so that they can grant you access. If you plan to keep using Heroku to host, you can use Fixie, the Heroku add-on, that will give you a static IP. This way, you won't have to

copy over their tables into your own database. You will also have access to everything in their database; including image and video attachments for the news and webinars, which are typically stored in the wp/uploads folder directly on their server.

2. I did not have the space in my database on Heroku's free platform to copy over the 'wp_users' table, so I am not retrieving authors for news and webinars. You will probably need to do this!

3. Most of my time was spent building out the features, so error handling and security might be lacking in places.

Let me know if you have more questions, and contact me so I can hand you over ownership of the Github repo and Heroku app!