Project Updates:

- Updated ER diagram and relational schema
- Set up postgres database with dummy data
- Set up web app, hosted on server
- Set up server-side API
- Set up basic querying to database w/ http requests

Open-Ended Questions:

- What are some must-haves for security protocol in our backend/database?
- Would it be possible for us to see the format of the data that Janna shared with you (file type, column headings, etc.)? This would allow us to create a method that she can use to upload the data herself.
- In phase 4 of the project, what do you mean by "the use of indices"?
- If the end goal is to deliver a product, should we prioritize simplicity and make sure our client has a working solution?
- What are some good database design principles that would help us avoid manually correcting every entry? Is there a way to "mass-change" a field?

Relational Schema Updates:

ENTITIES

```
CREATE TABLE Courses(
        scheduleNum INTEGER,
        course_num character varying (20) NOT NULL,
        semester character varying (6)  NOT NULL,
        lang character varying (15)  NOT NULL,
        program character varying (4)  NOT NULL,
        mum_of_conversation  integer NOT NULL,
        num_of_tutorials integer NOT NULL,
        academic_year integer NOT NULL,
        course_name  character varying (20)  NOT NULL,
        PRIMARY KEY (course_num, semester,academic_year)
 )
```

```
CREATE TABLE Instructors (
   email character varying NOT NULL,
   first_name character varying NOT NULL,
   last_name character varying  NOT NULL,
   preferred_mame character varying ,
   pronouns character varying ,
   role character varying  NOT NULL,
   academic_career character varying,
   languages_taught character varying NOT NULL,
   campus character varying NOT NULL,
   phone bigint NOT NULL,
   Graduation_Year bigint,
   Approved_to_Hire boolean,
   Paperwork_Status characterverying,
   Notes text,
   Hiring_History text
   PRIMARY KEY (email)
);


CREATE TABLE Students (
   email character varying(100) NOT NULL,
   first_name  character varying(50) NOT NULL,
   last_name  character varying(50) NOT NULL,
   preferred_name  character varying(50) NOT NULL,
   pronouns  character varying(50),
   campus  character varying NOT NULL,
   enrollment_history  character varying NOT NULL,
   academic_career  character varying(50) NOT NULL,
   graduation_year  character varying(255) NOT NULL,
   Other_Notes TEXT(500),
   Phone character varying(50),
   PRIMARY KEY (Email),
  );
```

RELATIONSHIPS

```
CREATE TABLE isRegistered(
         course_num  character varying(20) NOT NULL,
         semester  character varying(6) NOT NULL,
         academic_year  integer  NOT NULL,
         status  character varying(50) NOT NULL,
         email character varying(100) NOT NULL
         FOREIGN KEY (course_num)
             REFERENCES Courses,
         FOREIGN KEY (academic_year)
             REFERENCES Courses,
         FOREIGN KEY (semester)
             REFERENCES Courses,
          FOREIGN KEY (email)
            REFERENCES (students)
         PRIMARY KEY (course_num,semester,academic_year, email)
  )

CREATE TABLE isEnrolled(
         course_num character varying(20) NOT NULL,
         semester character varying(6) NOT NULL,
         academic_year integer NOT NULL,
         status character varying(50)  NOT NUL,
         email character varying(100) NOT NULL
         FOREIGN KEY (course_num)
             REFERENCES Courses,
         FOREIGN KEY (academic_year)
             REFERENCES Courses,
         FOREIGN KEY (semester)
             REFERENCES Courses,
          FOREIGN KEY (email)
            REFERENCES (students)
         PRIMARY KEY (course_num,semester,academic_year, email)
  )



CREATE TABLE participatesIn(
         role  character varying(20) NOT NULL,
         email character varying(100) NOT NULL,
         day character varying, NOT NULL,
         s_time time w/o timezone, NOT NULL,
         e_time time w/o timezone, NOT NULL,
            location  character varying , NOT NULL,
            campus character varying, NOT NULL,
            FOREIGN KEY (email)
                   REFERENCES Students,
            FOREIGN KEY (day)
                   REFERENCES Meetings,
              FOREIGN KEY (time)
                  REFERENCES (Meetings)
            FOREIGN KEY (location)
                  REFERENCES (Meetings)
            PRIMARY KEY (email,day,location,campus, e_time, s_time)


CREATE TABLE Leads(
            role character varying(20) NOT NULL,
            email character varying(100) NOT NULL,
            day character varying NOT NULL,
            s_time time w/o timezone NOT NULL,
            e_time time w/o timezone NOT NULL
            campus character varying NOT NULL,
            location character varying NOT NULL,
            FOREIGN KEY (email)
                   REFERENCES Instructors,
            FOREIGN KEY (day)
                   REFERENCES Meetings,
              FOREIGN KEY (date)
                  REFERENCES (Meetings)
            FOREIGN KEY (location)
                  REFERENCES (Meetings)
            PRIMARY KEY (email,day,location,s_time, e_time, campus)
)



CREATE TABLE taughtBy(
            role character varying(20) NOT NULL
            course_num character varying(20) NOT NULL
            semester character varying(6) NOT NULL
            academic_year INTEGER NOT NULL
```

```
        email character varying(100) NOT NULL
        FOREIGN KEY (course_num)
                REFERENCES Courses
        FOREIGN KEY (semester)
                REFERENCES Courses
        FOREIGN KEY (academic_year)
                REFERENCES Courses
        FOREIGN KEY (email)
                REFERENCES Instructors
        PRIMARY KEY ( course_num, semester, academic_year, email)



CREATE TABLE SpireStatus(
        email character varying(11) NOT NULL,
        course_num character varying(20) NOT NULL,
        status character varying NOT NULL,
        PRIMARY KEY (email, course_num),
        FOREIGN KEY (email) REFERENCES Students,
        FOREIGN KEY (course_num) REFERENCES Courses
)


CREATE TABLE When&Where (
        course_num character varying(20) NOT NULL,
        semester character varying(6) NOT NULL,
        Academic_year integer NOT NULL,
        day character varying , NOT NULL,
        s_time time w/o timezone NOT NULL,
        e_time time w/o timezone NOT NULL
        location character varying(100) NOT NULL,
        campus character varying NOT NULL,
        FOREIGN KEY (courseNum) REFERENCES Courses,
        FOREIGN KEY (academicYear) REFERENCES Courses,
        FOREIGN KEY (semester) REFERENCES Courses,
        FOREIGN KEY (day) REFERENCES Meetings,
        FOREIGN KEY (time) REFERENCES Meetings,
        FOREIGN KEY (location) REFERENCES Meetings,
        PRIMARY KEY (campus,day, s_time, e_time, location))
```

Sample Queries:

Which students still need to be interviewed?

```
SELECT public.students.first_name, public.students.last_name, public.is_enrolled.email,
public.is_enrolled.course_num FROM public.students, public.is_enrolled

WHERE public.students.email = public.is_enrolled.email AND public.is_enrolled.status = 'NX'
```

What classes are being held at UMASS on Tuesday? (This query will be adaptable depending on which campus/day/meeting type the user is looking for)

```
SELECT * FROM when_where
where campus = 'UMass' AND day = 'Wednesday' ;
```

Who are the instructors that have previously been conversational partners?

```
SELECT email FROM public.instructors
WHERE role = 'Conversation Partner' ;
```