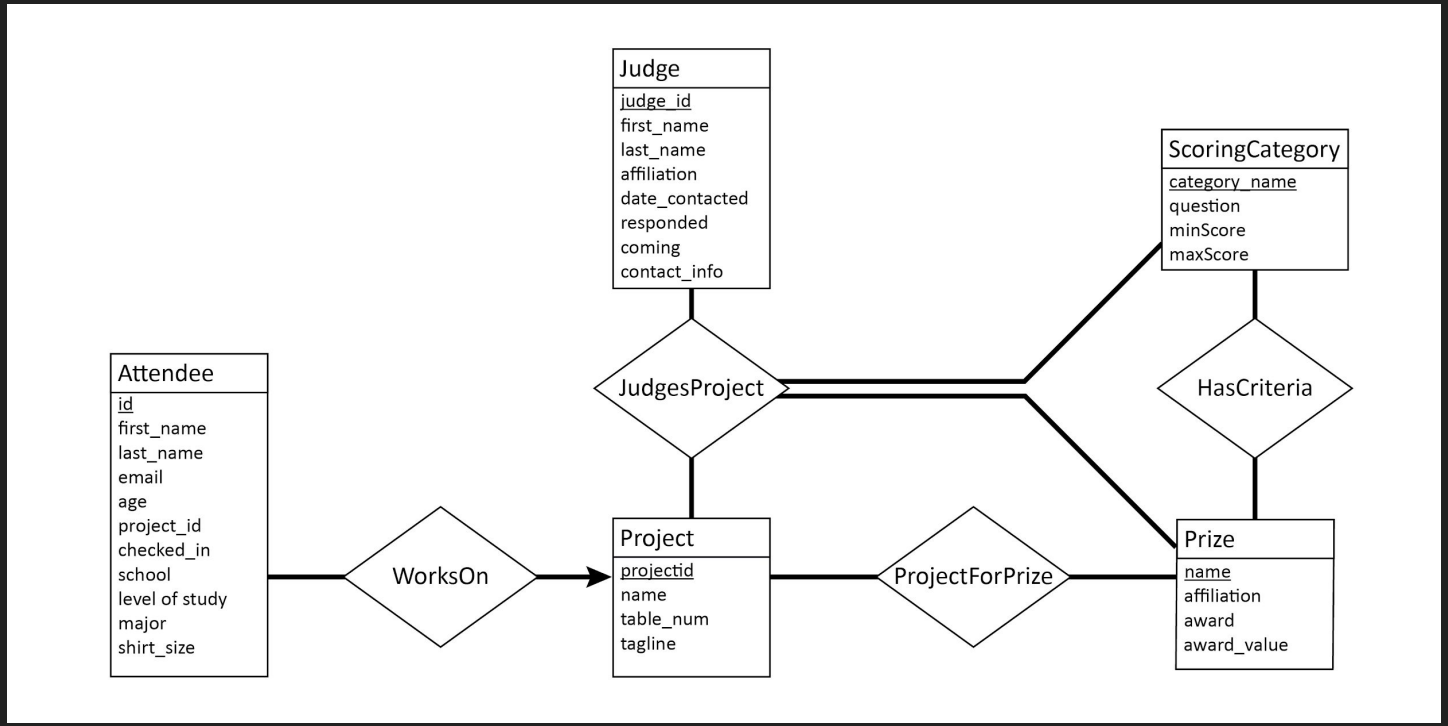
Hackathon Management System

Barry McCoy, Chuck Rakaczky, Ben Young

- HackCWRU needs a unified management system and website
  - Registration portal
  - View attendee information
    - Shirt sizes
    - Dietary restrictions
  - Check-in
  - Judging assignments
  - Project prize submissions
  - Table assignments
  - Scoring

ER Diagram


Judge
judge_id
first_name
last_name
affiliation
date_contacted
responded
coming
contact_info
ScoringCategory
category_name
question
minScore
maxScore
Attendee
id
first_name
last_name
email
age
project_id
checked_in
school
level of study
major
shirt_size
JudgesProject
HasCriteria
WorksOn
Project
projectid
name
table_num
tagline
ProjectForPrize
Prize
name
affiliation
award
award_value

Attendee

```
create table Attendee
(
    attendee_id    int         not null
        primary key,
    first_name     char(30)    null,
    last_name      char(30)    null,
    email          char(50)    null,
    age            int         null,
    project_id     int         null,
    `checked_in?`  tinyint(1)  null,
    school         char(100)   null,
    level_of_study char(100)   null,
    major          char(50)    null,
    shirt_size     char(30)    null,
    constraint Attendee_Project_project_id_fk
        foreign key (project_id) references Project (project_id)
            on update cascade on delete cascade
);
```

Judge

```
create table Judge
(
    judge_id        int         not null
        primary key,
    first_name      char(30)    null,
    last_name       char(30)    null,
    affiliation     char(100)   null,
    date_contacted  date        null,
    `responded?`    tinyint(1)  null,
    `coming?`       tinyint(1)  null,
    contact_info    char(50)    null
);
```

Project

```
create table Project
(
    project_id   int         not null
        primary key,
    project_name char(200) not null,
    table_num    int         null,
    tagline      text        null
);
```

Prize

```
create table Prize
(
    prize_name  char(100) not null
        primary key,
    sponsor     char(100) null,
    award       char(100) null,
    award_value int       null
);
```

ScoringCategory

```
create table ScoringCategory
(
    category_name char(30) not null
        primary key,
    question      text     not null,
    min_score     int      not null,
    max_score     int      not null
);
```

ProjectForPrize

```
create table ProjectForPrize
(
    project_id int       not null,
    prize_name char(100) not null,
    primary key (project_id, prize_name),
    constraint ProjectForPrize_Prize_prize_name_fk
        foreign key (prize_name) references Prize (prize_name)
            on update cascade on delete cascade,
    constraint ProjectForPrize_Project_project_id_fk
        foreign key (project_id) references Project (project_id)
            on update cascade on delete cascade
);
```

DietRestriction

```
create table DietRestriction
(
    attendee_id int       not null,
    restriction char(100) not null,
    primary key (attendee_id, restriction),
    constraint DietRestriction_Attendee_attendee_id_fk
        foreign key (attendee_id) references Attendee (attendee_id)
            on update cascade on delete cascade
);
```

HasCriteria

```
create table HasCriteria
(
    prize_name    char(100) not null,
    category_name char(30)  not null,
    primary key (prize_name, category_name),
    constraint HasCriteria_Prize_prize_name_fk
        foreign key (prize_name) references Prize (prize_name),
    constraint HasCriteria_ScoringCategory_category_name_fk
        foreign key (category_name) references ScoringCategory (category_name)
            on update cascade on delete cascade
);
```

JudgesProject

```
create table JudgesProject
(
    judge_id      int       not null,
    project_id    int       not null,
    prize_name    char(100) not null,
    category_name char(30)  not null,
    score         float     null,
    primary key (judge_id, prize_name, project_id, category_name),
    constraint JudgesProject_Judge_judge_id_fk
        foreign key (judge_id) references Judge (judge_id),
    constraint JudgesProject_Prize_prize_name_fk
        foreign key (prize_name) references Prize (prize_name)
            on update cascade on delete cascade,
    constraint JudgesProject_Project_project_id_fk
        foreign key (project_id) references Project (project_id)
            on update cascade on delete cascade,
    constraint JudgesProject_ScoringCategory_category_name_fk
        foreign key (category_name) references ScoringCategory (category_name)
            on update cascade on delete cascade
);
```

- MySQL
- Flask
- Real data from HackCWRU 2020 with fake names, scrambled info
  - Attendees
  - Judges
  - Judging results

Sample Query: Count each shirt size

```
# Get quantity of each shirt size
SELECT shirt_size, COUNT(*)
FROM Attendee
GROUP BY shirt_size
ORDER BY COUNT(*) desc
```

Sample Query: Project submitted for >1 Track

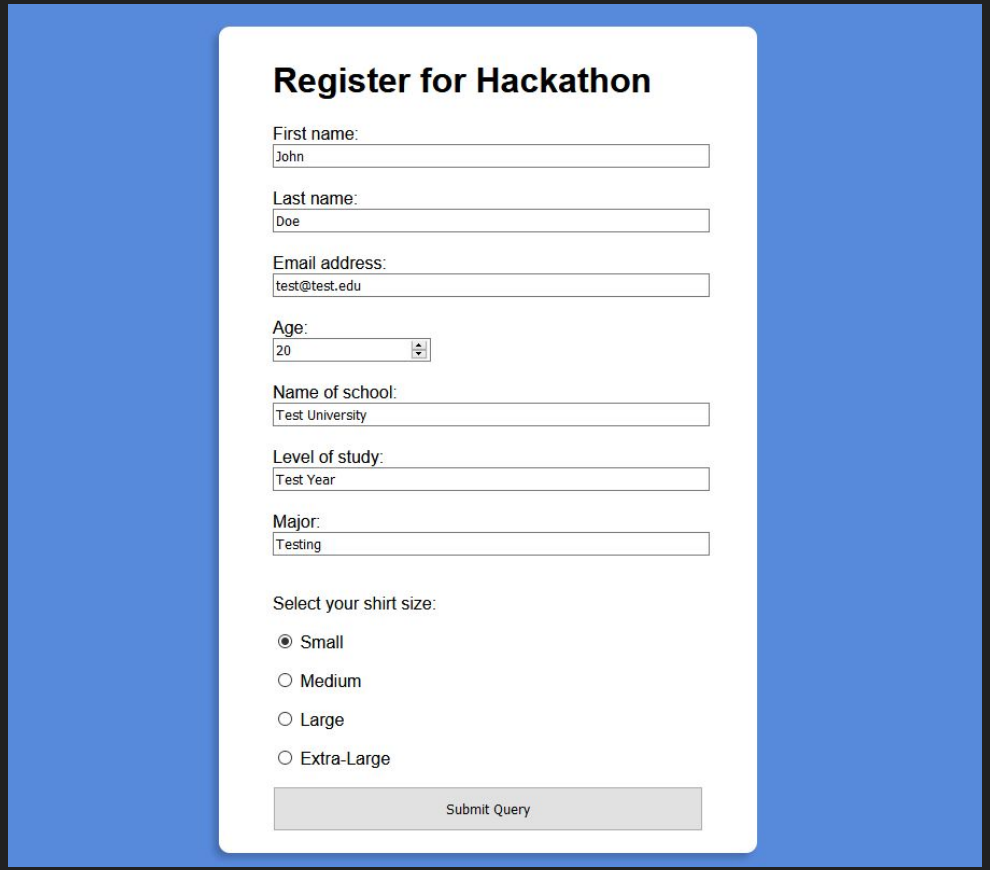
```
# Get names of projects that submitted to more than one track
SELECT Project.project_name, T.track_count
FROM Project JOIN (
    SELECT project_id, COUNT(*) as track_count FROM ProjectForPrize
    WHERE prize_name IN ("Health Track", "Maker Track", "Civic Track", "FinTech Track")
    GROUP BY project_id
    HAVING Count(*) > 1
    ) T
on Project.project_id = T.project_id
```

Sample Query: Get the winning project for each prize

```
WITH ScoreByPrize AS (
    SELECT name,
           AVG(score) AS p_score,
           prize
           # get total score for each project by judge
    FROM (
             SELECT P.project_name AS name, J.prize_name as prize, SUM(J.score) AS score
             FROM JudgesProject J,
                  Project P
             WHERE J.project_id = P.project_id
             GROUP BY P.project_id, J.judge_id, J.prize_name
         ) as ScoreByJudge
    GROUP BY name, prize
    ORDER BY AVG(score) desc
)
SELECT ScoreByPrize.prize, ScoreByPrize.name, max_p_score
FROM (
    ScoreByPrize
    JOIN
     (
         SELECT ScoreByPrize.prize,
                MAX(ScoreByPrize.p_score) AS max_p_score
         FROM ScoreByPrize
         GROUP BY ScoreByPrize.prize
     ) MaxScoreByPrize
     ON MaxScoreByPrize.max_p_score = ScoreByPrize.p_score
         AND MaxScoreByPrize.prize = ScoreByPrize.prize
    )
```

Screenshots

Registering



Code behind registration (how we call our queries)

```
35  #@app.route('/register/<error>')
36  @app.route('/register/', methods=['POST', 'GET'])
37  def register(error = None):
38      error = None
39      if request.method == 'POST':
40          if valid_registration(request.form):
41              register_attendee(request.form)
42              return confirmation(request.form['firstname'])
43          else:
44              error = "Invalid registration"
45
46      return render_template('attendee_register.html', error = error)
47
48  @app.route('/confirmation/<name>')
49  def confirmation(name=None):
50      return render_template('registration_confirmation.html', name = name)
51
52
53  def valid_registration(registrationForm):
54      #registrationForm['firstname']
55      # for now, just to see if I can insert an attendee successfully
56      return True
57
58  def register_attendee(form):
59      with open('../get_max_attendee_id.sql', 'r') as file:
60          getMaxId = file.read()
61          cur = mysql.connection.cursor()
62          cur.execute(getMaxId)
63          maxId = cur.fetchone()
64          cur.execute('''INSERT INTO attendee(attendee_id, first_name, last_name, email, age, project_id, checked_in,
65          mysql.connection.commit()
```

Remaining Work

Assigning Judges to Projects

Creating administrative pages for organizers

Creating project-related pages for users

Creating non-essential pages that are nice for users to have