

# THE Internet Explorer's Cafe Seating and Reservation System

...

**THE Internet Explorers**

Nick Harvey + Alli Hornyak + Emily Niehaus + Tze Hei Tam + Nina Yao

# Concept

The Seating and Reservation System was designed to accommodate both **employees** and **customers**, granting them the ability to **create** and **edit** reservations.

From the Host Dashboard, an employee can see the **queue** of reservations for that day as well as the table **layout** and **availability**.

The design is suitable for a variety of restaurants; the layout image is easily swapped and table information is **flexible** in creation.

The Internet Explorers Cafe: Seating Management System

[Edit Account](#) [Logout](#)

### Current Seating

[View Seating Map](#)

Table	Type	Accessible?	# Seats	Status:
1	Booth	Yes	4	Assigned <a href="#">Unassign</a>
2	Booth	Yes	6	Available <a href="#">Assign</a>
3	Table	No	4	Available <a href="#">Assign</a>
4	Hightop	No	3	Available <a href="#">Assign</a>

### Queue For Today

Arrival Time	Party	Size	Requests	Phone
--------------	-------	------	----------	-------

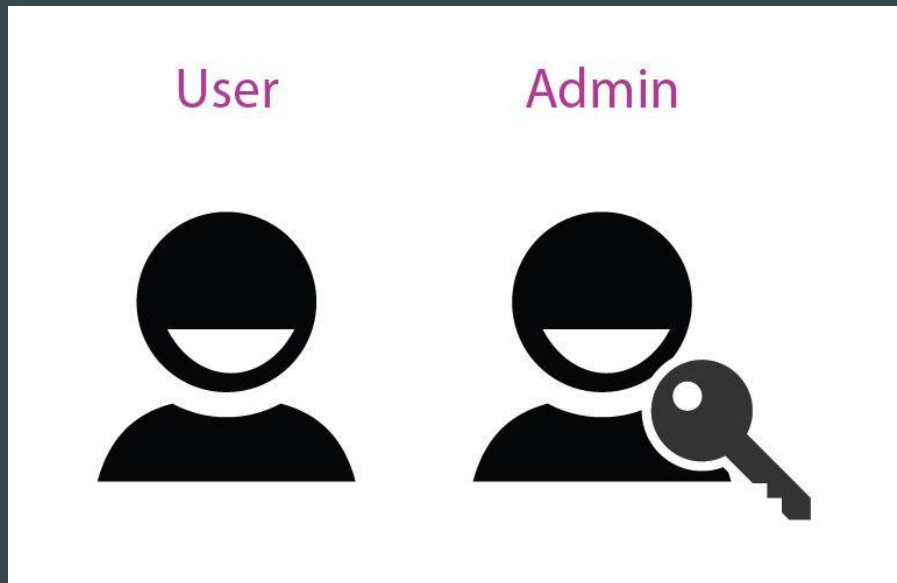
### All Reservations

[Add New](#)

Reservation Time	Party	Size	Requests	Phone	
2020-04-24 03:00	Nina Today	4	None	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-24 05:00	Alii Today	3	Wheelchair Accessible	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-24 06:00	Alii Today	3	Allergic to gluten	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-24 07:00	Tze-day	10	None	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-24 12:00	Emily Today	7	None	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-24 14:00	Tze Hui Today	1	None	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-24 22:00	Nick Today	3	Wheelchair Accessible	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>
2020-04-30 01:54	Bubblers	5	Batman's please	1234567890	<a href="#">Edit</a> <a href="#">Delete</a>

# User System

- Differentiates between **customers** and **admins**, and provides **separate views** for both
- Both types of users are derived from the **same model and controller** with much of the functionality provided by **Devise**
- Admin assigned by a **boolean** in the **schema**



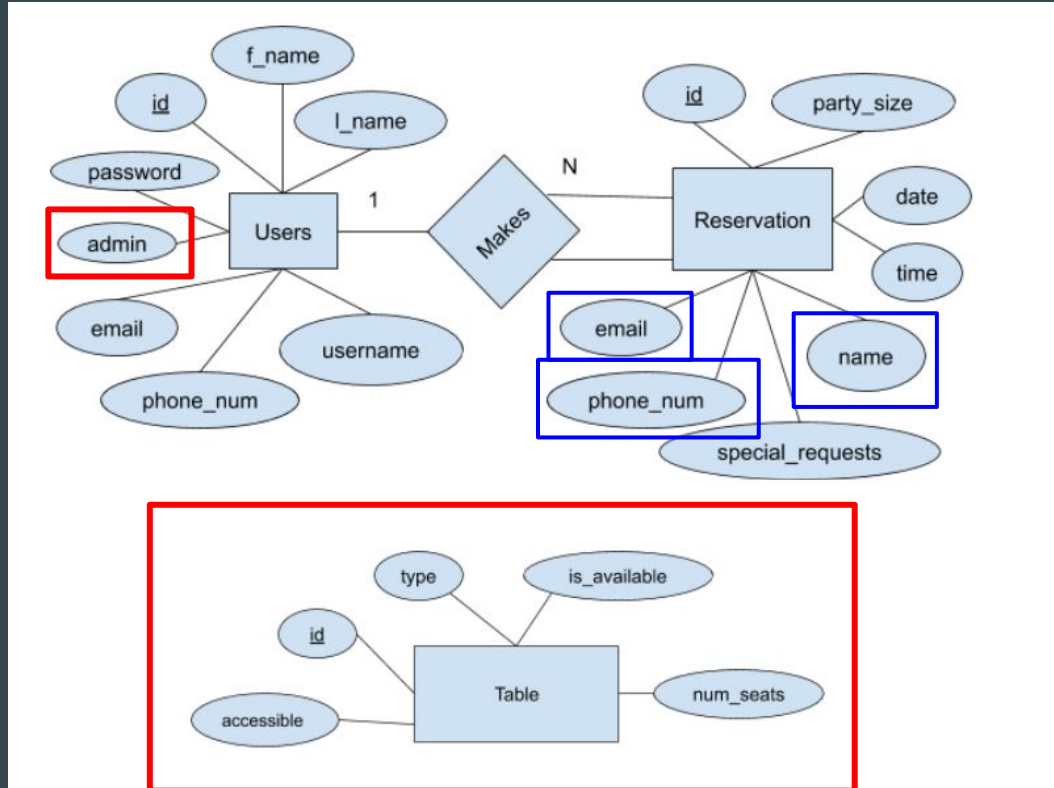


Demo



# Database Schema

# Entity Relationship Diagram



# User Model

- Used **Devise** for user login

```
create_table "users", force: :cascade do |t|
  t.string "email", default: "", null: false
  t.string "encrypted_password", default: "", null: false
  t.string "reset_password_token"
  t.datetime "reset_password_sent_at"
  t.datetime "remember_created_at"
  t.string "f_name"
  t.string "l_name"
  t.string "phone_num"
  t.boolean "admin"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
  t.index ["email"], name: "index_users_on_email", unique: true
  t.index ["reset_password_token"], name:
"index_users_on_reset_password_token", unique: true
end
```



# User Model (Continued)

- Used **Devise** for user login

```
class User < ApplicationRecord
  has_many :reservations
  # Include default devise modules. Others available are:
  # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
  before_save { self.email = email.downcase }
  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :validatable, password_length: 8..128
  validates :phone_num, :f_name, :l_name, presence: true

  VALID_PHONE_REGEX = /\d{10}/
  VALID_EMAIL_REGEX = /\S+@\S+\.\S+/
  validates :phone_num, length: {minimum: 10, maximum: 10}, format: { with:
VALID_PHONE_REGEX}
  validates :email, uniqueness: true, length: { maximum: 255 }, format: {with:
VALID_EMAIL_REGEX}
end
```



# Reservation Model

```
create_table "reservations", force: :cascade do |t|
  t.integer "party_size"
  t.time "time"
  t.date "date"
  t.string "special_requests"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
  t.integer "user_id"
  t.string "name"
  t.string "phone_num"
  t.string "email"
  t.index ["user_id"], name: "index_reservations_on_user_id"
end
```

# Reservation Model (Continued)

- Used **regular expressions** for validating **phone numbers** and **emails**

```
class Reservation < ApplicationRecord
  belongs_to :user
  default_scope -> { order(:date, :time) }
  validates :date, :time, :party_size, :name, :phone_num, :email, presence:
true
  VALID_PHONE_REGEX = /\d{10}/
  validates :phone_num, length: {minimum: 10, maximum: 10}, format: { with:
VALID_PHONE_REGEX}
  VALID_EMAIL_REGEX = /\S+@\S+\.\S+/
  validates :email, format: { with: VALID_EMAIL_REGEX}
end
```

# Table Model

```
create_table "tables", force: :cascade do |t|
  t.string "table_type"
  t.integer "num_seats"
  t.boolean "is_available"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
  t.boolean "accessible"
end
```

- Every table **required** a **type** and **number of seats**

```
class Table < ApplicationRecord
  validates :table_type, :num_seats, presence: true
end
```



# Architecture

# Routes

```
Rails.application.routes.draw do
  devise_for :users
  resources :reservations
  resources :tables, only: [:update]

  # For details on the DSL available within this file, see http://guides.rubyonrails.org/routing.html

  root to: 'pages#home'
end
```

## The Internet Explorers Cafe: Seating Management System

Campus cafe located in **THE** heart of Ohio States University District.  
Live music, hot coffee, great bean selection and a food menu to please everyone!

### Hours

**Mondays**...Closed

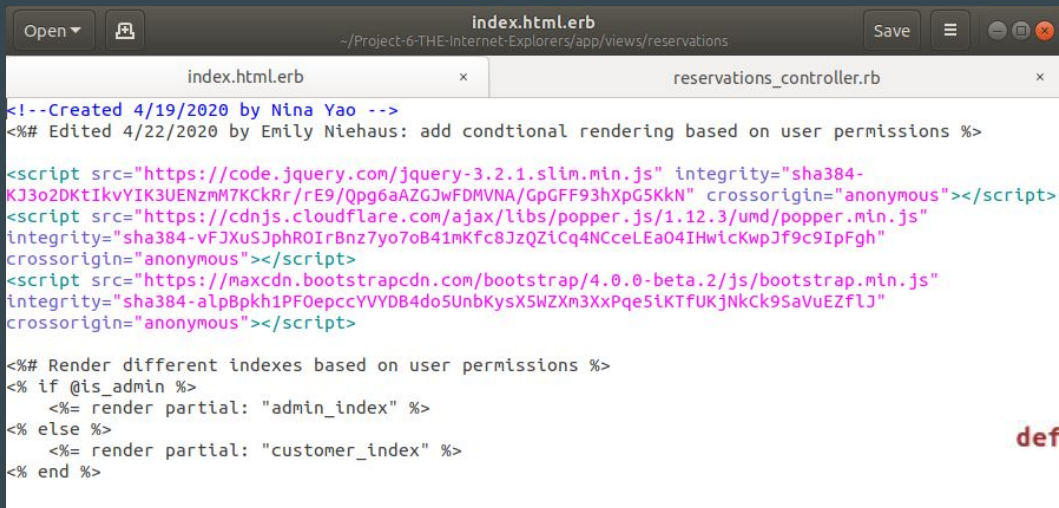
**Tuesday - Saturday**...7pm - 2:30am

**Sunday**...Browns games when in season

**GAMEDAYS**...Hours vary, check our social media

**Closed** most holidays but **ALWAYS OPEN** for New Year's Eve

# Index



```
index.html.erb
x
reservations_controller.rb
x

<!--Created 4/19/2020 by Nina Yao -->
<## Edited 4/22/2020 by Emily Niehaus: add conditional rendering based on user permissions %>

<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-
KJ3o2DKtIkvyIK3UENzmM7KcKRR/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.3/umd/popper.min.js"
integrity="sha384-vfJXuSjphR0IRBnz7yo7oB41mKfc8JzQZiCq4NCceLEa04IHwicKwpJf9c9IpFgh"
crossorigin="anonymous"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta.2/js/bootstrap.min.js"
integrity="sha384-alpBpkh1PFOepccYVYDB4do5UnbKysX5WZXM3XxPqe5iKTfUKjNkCk9SaVuEZflj"
crossorigin="anonymous"></script>

<## Render different indexes based on user permissions %>
<% if @is_admin %>
  <%= render partial: "admin_index" %>
<% else %>
  <%= render partial: "customer_index" %>
<% end %>
```

```
def index
  if current_user.admin == true
    @users = User.all
    @is_admin = true
    @tables = Table.all
    @queue = Reservation.where date: Date.today
    @reservations = Reservation.all
  else
    @users = [current_user]
    @is_admin = false
    @tables = Table.all
  end
end
```

# New Reservation

```
<%= button_to "Add New", new_reservation_path, method: 'get', class: "btn btn-default btn-primary" %>
```

## All Reservations

Add New

Reservation Time	Party	Size	Requests	Phone		
2020-04-24 12:00	Emily Today	7	none	1234567890	Edit	Delete
2020-04-24 14:00	Tze Hei Today	1	none	1234567890	Edit	Delete
2020-04-25 05:00	Bananas Hello	3	none	1234567890	Edit	Delete
2020-04-25 06:00	Alli HELLO	3	none	1234567890	Edit	Delete



# New Reservation

les [Back to Reservations](#)

## Create Reservation

Reservation Name:

Customer Email:

Customer Phone Number:

Reservation Date:

Reservation Time:  
 :

Party Size:

Special Requests:

[Create Reservation](#)

```
<%= button_to "Back to Reservations", reservations_path, method: 'get', class: "btn btn-default btn-primary" %>
```

```
<p>Party Size:<br> <%=f.nu  
<p>Special Requests:<br> <  
<%= f.submit %>
```

```
def create  
  @customer = current_user  
  @reservation = current_user.reservations.build(reservation_params)  
  if @reservation.date > Date.today or @reservation.time.hour > Time.now.hour  
  or(@reservation.time.hour == Time.now.hour and @reservation.time.min > Time.now.min)  
    if @reservation.save  
      redirect_to reservations_url  
    else  
      redirect_to new_reservation_url  
    end  
  else  
    redirect_to new_reservation_url  
  end  
end
```

# Edit Reservation

```
<%= button_to "Edit", edit_reservation_path(reservation), method: 'get', class: "btn btn-default btn-primary" %>
```

## All Reservations

[Add New](#)

Reservation Time	Party	Size	Requests	Phone		
2020-04-24 12:00	Emily Today	7	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>
2020-04-24 14:00	Tze Hei Today	1	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>
2020-04-25 05:00	Bananas Hello	3	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>
2020-04-25 06:00	Alli HELLO	3	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>

# Edit Reservation

Edit Reservation for

Enter name:

Enter email:

Enter phone number:

Enter date:

Enter time:  
 :

Enter party size:

Enter special requests:

```
def edit
  @reservation = Reservation.find(params[:id])
end

def update
  @reservation = Reservation.find(params[:id])
  if Date.parse(params[:reservation][:date]) > Date.today or
    params[:reservation]["time(4i)"].to_i > Time.now.hour or
    (params[:reservation]["time(4i)"].to_i > Time.now.hour and
     params[:reservation]["time(5i)"].to_i > Time.now.min)
    if @reservation.update(reservation_params)
      redirect_to reservations_url, notice: 'Success!'
    else
      redirect_to edit_reservation_path(params[:id])
    end
  else
    redirect_to edit_reservation_path(params[:id])
  end
end
```

# Delete Reservation

```
def destroy
  @reservation= Reservation.find(params[:id])
  @reservation.destroy()
  redirect_to reservations_url
end
```

```
<%= button_to 'Delete', reservation_path(reservation), method: 'delete', class: "btn btn-default btn-primary" %>
```

## All Reservations

[Add New](#)

Reservation Time	Party	Size	Requests	Phone		
2020-04-24 12:00	Emily Today	7	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>
2020-04-24 14:00	Tze Hei Today	1	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>
2020-04-25 05:00	Bananas Hello	3	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>
2020-04-25 06:00	Alli HELLO	3	none	1234567890	<a href="#">Edit</a>	<a href="#">Delete</a>

# Assigning Tables

```
<% if t.is_available %>
Available
<%= button_to "Assign", table_path(t), method: 'put', class: "btn btn-default btn-primary" %>
<% else %>
Assigned
<%= button_to "Dismiss", table_path(t), method: 'put', class: "btn btn-default btn-primary"%>
<% end %>
</span>
```

## Current Seating

[View Seating Map](#)

Table	Type	Accessible?	# Seats	Status:
1	booth	Yes	4	Assigned Dismiss
2	booth	Yes	6	Available Assign
3	table	No	4	Available Assign
4	hightop	No	3	Available Assign

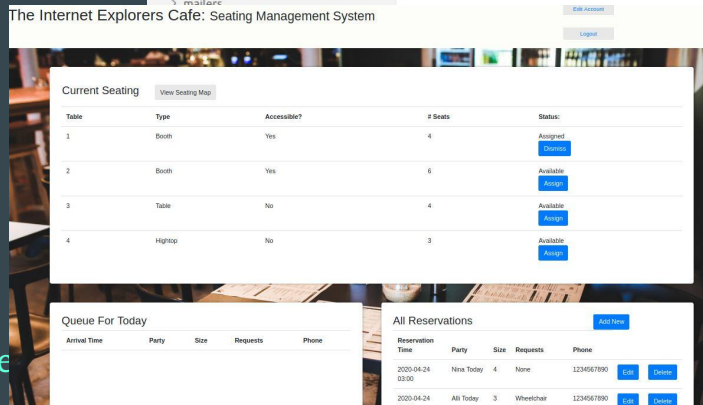
```
def update
  @table = Table.find(params[:id])
  @table.is_available = !@table.is_available
  @table.save

  redirect_to reservations_url
end
```



- A web application **framework**
  - The **core** of the application
  - Implements the **MVC** pattern
- Provides database table **creation**, **migration**, and **scaffolding** for views to structure a web page around
- Makes **assumptions** about the beginnings of a web site, allowing a developer to **write less code** while still **retaining functionality**

```
5 # Edited 4/22/2020 by Tze Hei Tam: Removed redundant routes
6 # Edited 4/23/2020 by Tze Hei Tam: Removed routes from tables
7
8 Rails.application.routes.draw do
9   devise_for :users
10  resources :reservations
11  resources :tables, only: [:update]
12
13  # For details on the DSL available within this file, see ht
14
15  root to: 'pages#home'
16 end
17
```





- A gem used for **user authentication**
  - Provides **session handling** and **identity verification**
  - Integral to the user model and validating its attributes
- Used with Rails to generate the **user views**, such as **login** and **sign up**
  - Provides helper methods such as **current\_user** to access user information for the current session

```
class User < ApplicationRecord
  has_many :reservations
  # Include default devise modules. Others available are:
  # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
  before_save { self.email = email.downcase }
  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :validatable, password_length: 8..128
  validates :phone_num, :f_name, :l_name, presence: true

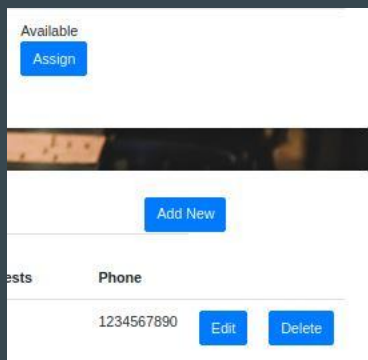
  VALID_PHONE_REGEX = /\d{10}/
  VALID_EMAIL_REGEX = /\S+@\S+\.\S+/
  validates :phone_num, length: {minimum: 10, maximum: 10}, format: {with: VALID_PHONE_REGEX}
  validates :email, uniqueness: true, length: {maximum: 255}, format: {with: VALID_EMAIL_REGEX}
end
```

```
views
└─ devise
   ├── confirmations
   ├── mailer
   ├── menu
   ├── passwords
   └─ registrations
      ├── edit.html.erb
      ├── new.html.erb
      └─ sessions
         ├── shared
         └─ unlocks
```





- Primarily used for **reactive UI**
  - Allows you to define the **state** of different elements, **updating** and **rendering** as these states change
  - Prevents a page from reloading
  - Provides a **smoother** user **experience**



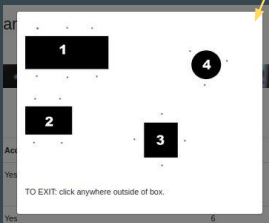
- Foundation of buttons which actively **change** the **database** and **show** a **response**
  - **Assign** and **Delete** buttons
  - **No refresh** required to **reflect change**

```
class Header extends React.Component {
  constructor(props) {
    super(props);
    this.state = { is_avail: true };
  }

  render() {
    if (!this.state.is_avail) {
      return React.createElement("div", null,
        React.createElement("p", {class: "d-inline"}, "Occupied"),
        React.createElement(
          "button", {
            onClick: () => this.setState({ is_avail: true }),
            class: "d-inline ml-3"},
            "Dismiss"));
    }
    else {
      return React.createElement("div", null,
        React.createElement("p", {class: "d-inline"}, "Available"),
        React.createElement(
          "button", {
            onClick: () => this.setState({ is_avail: false }),
            class: "d-inline ml-3"},
            "Assign"));
    }
  }
}
```

# Bootstrap

- A gem used for **front-end development**
  - A library of **CSS, HTML** and **Javascript** code used to create modern, responsive applications
- Provided high quality **components** which eased the creation of web pages
- Used throughout the application for **styling** and **functionality**



Current Seating View Seating Map

Table	Type	Accessible?	# Seats	Status:
1	Booth	Yes	4	Assigned <a href="#">Details</a>
2	Booth	Yes	6	Available <a href="#">Assign</a>
3	Table	No	4	Available <a href="#">Assign</a>
4	Hightop	No	3	Available <a href="#">Assign</a>

```
<div class="container-fluid">
  <div class="row mr-5 ml-5">
    <%% CURRENT SEATING TABLE %%>
    <div class="col-md-12 main">
      <div class="row">
        <div>
          <h3 class="d-inline m-3">
            Current Seating
          </h3>
          <button type="button" class="btn d-inline m-3" data-toggle="modal" data-target="#myModal">
            View Seating Map
          </button>
        </div>
        <div id="myModal" class="modal fade" tabindex="-1" role="dialog" aria-labelledby="myModalLabel">
          <div class="modal-dialog">
            <div class="modal-content">
              <div class="modal-body">
                
                <p>TO EXIT: click anywhere outside of box.</p>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
```