Project Step 7 - Portfolio Assignment

Contacts Webapp

http://flip3.engr.oregonstate.edu:11395/
(OSU VPN)

Project Group 19 - George Hill & Jacky Tran

CS340 - Section 401

Submitted March 19, 2020

Icon made by Freepik from www.flaticon.com

Table of Contents

Executive Summary

This project is a multi-user contact manager whose unique feature is the ability to log interactions with contacts. This website allows users to create contacts, specify the days they interacted with each contact, and record the details of several interactions per day.

The biggest change to this project came at the very beginning. Our initial proposal was to create a contacts manager that would let users track, for each contact, a variable number of email addresses and a variable number of phone numbers (as separate entities). It would also allow a user to log interactions with one or more contacts (e.g. group calls). We were advised by the professor, however, to use the following entities: contacts, interactions, interaction details, and communication modes. We were also advised as to what attributes each entity should have and as to what relationships should exist between entities. We have implemented the professor's suggestions, with only a few modifications.

One such modification came as a result of the peer feedback we received from our draft HTML interface design. Several classmates remarked that they could not find a nullable one-to-many relationship, and it was indeed a detail that had been overlooked. We set upon the simple solution of adding, to our "contacts" entity, an "emergencyContactId" attribute that uses another contact's ID as a foreign key, creating a nullable one-to-many relationship within the "contacts" entity.

We also received several pieces of UI advice from our peers. For example, a classmate noted that it was unclear in our HTML design if a user was logged in or not. That led to another easy but necessary fix, and the website now displays the user’s name in the nav bar after logging in.

The next important piece of feedback came during the draft SQL queries step. One peer suggested that we use more ON DELETE statements on foreign keys to prevent errors. We put the suggestion on the back burner while we started working on the backend, only for that classmate to be proven correct when we ran into errors trying to delete contacts that had foreign key dependents in the interactions table. Consequently, we added multiple ON DELETE statements to our data-definition queries.

Other changes were generally minor. At one point, for example, we realized that we needed to add "detailsId" as a primary key for our "interactionDetails" entity so that we could make those relationships deletable. Likewise, our web interface went through some revisions. For example, users originally "signed in" by using a drop-down menu. However, we realized that new users might not realize that they need to hover over it to activate it, and so we replaced it with a table.

Project Outline

Overview

This Contacts Webapp is similar to applications such as the contacts app on your Android or iOS device. It lets users record basic contact information about people they know. Its unique feature solves a problem that is not addressed, however, by most other contacts apps: it lets the user track interactions with each contact.

Interaction logging is commonly needed in law firms, schools, businesses, non-profits, and healthcare settings, where people need to have a record of the kinds of interactions they have had with their various contacts. These records typically include information about the date, the time, and the nature of each interaction.

The way that this application is structured, each interaction is with a particular contact on a particular day. Each interaction in turn has details that specify the mode of communication (e.g. voice call, SMS, in-person meeting), the start time of the interaction, and a description of the nature of the interaction (e.g. "Met to discuss CS project").

Our implementation of this application is ideal for small-scale usage, i.e. tens of users, each with tens of contacts and interactions. While the website could support more, the UI is not designed to scale for a large user base or power users.

Detailed Implementation Walkthrough

The "users" entity has the following attributes: "userId", "lastName", "firstName", "phone", and "email". The application allows users to update their name, phone, and email information. The application starts populated with sample users and allows more to be added. There is no particular limit set on the number of users, but it would be ungainly to have more than thirty or so, due to the limitations of the user interface. The "userId" attribute is a primary key.

The "contacts" entity has the following attributes: "contactId", "userId", "lastName", "firstName", "phone", "email", "notes", and "emergencyContactId". The name, phone, email, and notes attributes are used to track basic contact information. The "userId" attribute holds a foreign key that matches the "userId" attribute of the "users" entity that owns the contact. The "emergencyContactId" attribute also holds a foreign key, but it is a nullable foreign key that, if present, matches the "contactId" of another contact. The application starts with about six sample contacts. It allows contacts to be added, edited, and deleted. Again, there is no particular limit on the number of contacts per user, but there are UI limitations. The list of contacts can be filtered by running "contains" searches against "firstName" and/or the "lastName" attributes. The "contactId" attribute is a primary key.

The "communicationModes" entity has the following attributes: "comId" and "type". The "type" attribute stores terms describing different modes of communication. The application starts with "SMS Message", "Voice Call", and "Meeting". Users can add additional modes. As before, there is no particular limit on the number of modes, but it would be ungainly to try to manage more than about thirty. The "comId" attribute is a primary key.

The "interactions" entity has the following attributes: "interactId", "contactId", and "startDate". The "startDate" attribute contains the date of the interaction. The "contactId" attribute contains a foreign key that refers to the "contactId" of a record in the "contacts" table; this indicates which contact the user had the interaction with. The database starts with several sample interactions for several of the contacts. The user can add new interactions. As before, having more than thirty or so would be ungainly. The "interactId" attribute is a primary key.

The "interactionDetails" entity has the following attributes: "detailsId", "interactId", "comId", "startTime", and "details". The "startTime" attribute contains the time of the interaction. The "details" attribute contains a note describing the nature of the interaction (e.g. "Met to discuss CS project"). The "comId" attribute contains a foreign key that matches the "comId" of one of the "communicationModes" entities; this indicates the mode of the communication (e.g. "SMS Message", "Voice Call", "Meeting"). The "interactId" attribute contains a foreign key that matches the "interactId" of one of the "interactions" entities; this indicates which interaction this record belongs to. The database starts with about nine sample "interactionDetails" entities, spread amongst several sample interactions (and, hence, sample contacts and sample users). The user can add and delete "interactionDetails" entities. The "detailsId" attribute is a primary key; it is necessary so that these kinds of entities can be deleted.

Navigating the website begins at the home page, where a user can click on an existing user or can add a new user. The user is then brought to the "myAccount" page, where the user is shown the user's currently on-file contact information; that can updated if necessary. From that page, the user clicks on the "Contacts" menu option to go to the "contacts" page, where the user is shown a list of all of the user's contacts. The user can filter the list using the text inputs at the top of the page; the user can add a new contact using the form at the bottom of the page, and the user can edit or delete contacts using the buttons on the table of contacts.

One of the buttons on the "contacts" page is the "Interactions" button, which takes the user to the "interactions" page for a particular contact. This page lists all existing interactions and allows the user to create new interactions; interactions are listed as dates. Clicking on the "View Details" button then takes the user to the "interactionsDetails" page for a particular interaction (i.e. particular date). That page lists all times that the user had particular forms of interaction, accompanied by detailed notes (e.g. 9:00 PM, SMS Message, "Wished him a happy New Year." On that page, the user can add new and delete interaction details for that date.

Additionally, there is a "modes" page, where the user can see and add to the list of communication modes. To log out, or to switch users, the user clicks on the "Home" menu option.

Database Outline

users: records the details of the **users** of this Contact Webapp

- *userId*: int, PK, auto_increment, not NULL
- *lastName*: varchar(50), not NULL
- *firstName*: varchar(50), not NULL
- *phone*: varchar(50), not NULL
- *email*: varchar(254)[1], not NULL
- Relationship: 1:M between **users** and **contacts** with *userID* as a FK inside of **contacts**

contacts: records the details of the **contacts** for each **user**

- *contactId*: int, PK, auto_increment, not NULL
- *userId*: int, not NULL, FK
- *lastName*: varchar(50), not NULL
- *firstName*: varchar(50), not NULL
- *phone*: varchar(50), not NULL
- *email*: varchar(254), not NULL
- *notes*: varchar(255), not NULL
- *emergencyContactId*: int, FK, default NULL
- Relationship: 1:M between **contacts** and **interactions** with *contactID* as a FK inside of **interactions**
- Relationship: 1:M between **contacts** and itself with *emergency_contactId* as a FK inside of **contacts**.

interactions: records each instance of an **interaction** between a **user** and a **contact**

- *interactId*: int, PK, auto_increment, not NULL
- *contactId*: int, not NULL, FK
- *startDate*: date, not NULL
- Relationship: 1:M between **interactions** and **interactionDetails** with *interactId* as a FK inside of **interactionDetails**
- Relationship: M:M between **interactions** and **communicationModes** facilitated by **interactionDetails**. Each **interaction** will be able to be tagged with one or more **communicationModes** (i.e. call, email, SMS, or meeting). In turn, each of those **communicationModes** will appear in, potentially, many different **interactions**.

[1] See: https://stackoverflow.com/questions/7717573/what-is-the-longest-possible-email-address

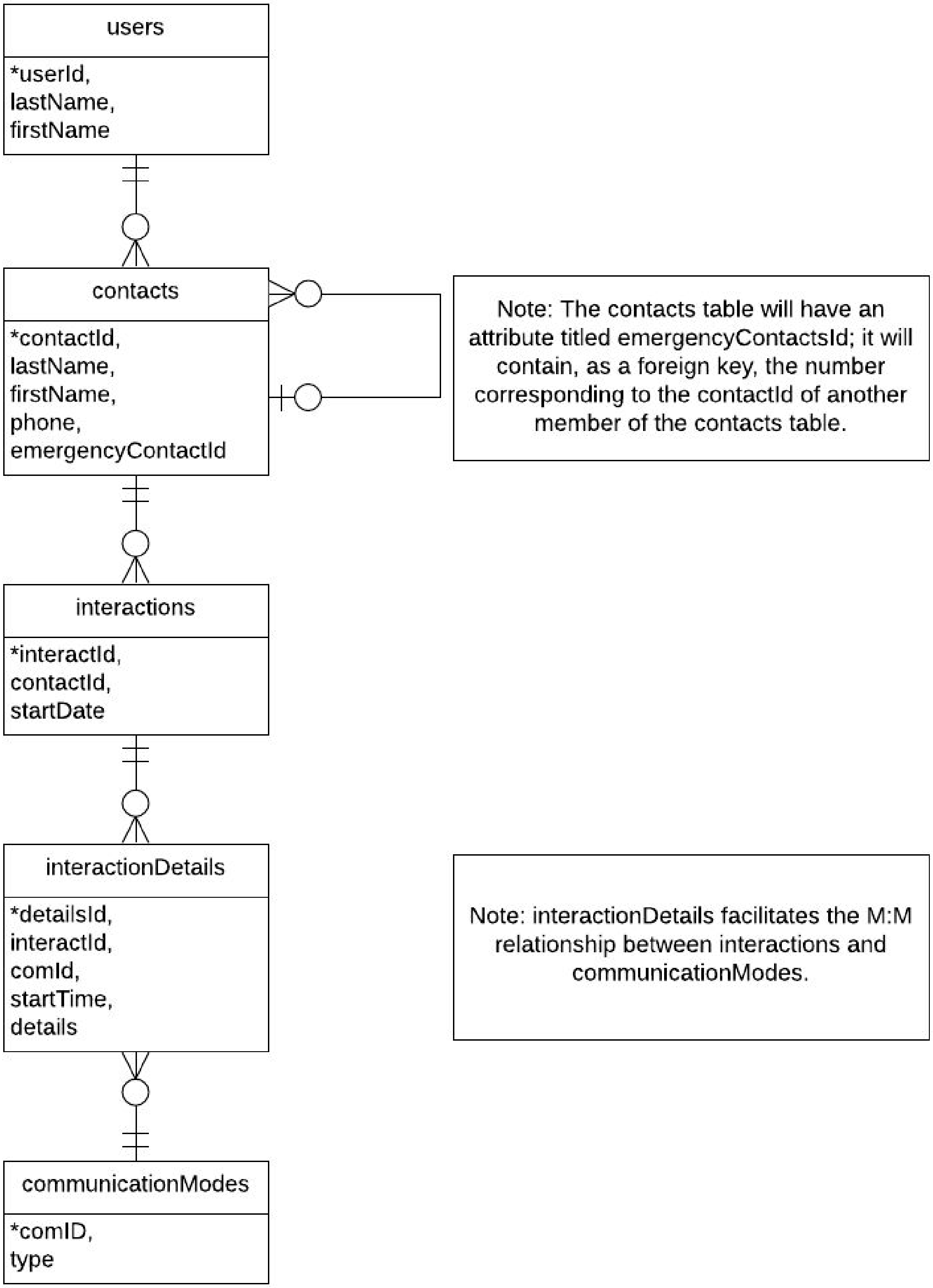
interactionDetails: records the details of each **interaction**

- *detailsId:* int, PK, auto_increment, not NULL
- *interactId*: int, not NULL, FK
- *comId*: int, not NULL, FK
- *startTime*: time, not NULL
- *details*: varchar(255), not NULL
- Relationship: M:1 between **interactionDetails** and **interactions**
- Relationship: M:1 between **interactionDetails** and **communicationModes**

communicationModes: records the **communicationModes** possible between **users** and **contacts**

- *comId*: int, PK, auto_increment, unique, not NULL
- *type*: varchar(50), not NULL
- Relationship: 1:M between **communicationModes** and **interactionDetails** with *comId* as a FK inside of **interactionDetails**
- Relationship: M:M between **communicationModes** and **interactions** facilitated by **interactionDetails**. As noted above, each **interaction** will be able to be tagged with one or more **communicationModes** (i.e. call, email, SMS, or meeting). In turn, each of those **communicationModes** will appear in, potentially, many different **interactions.**

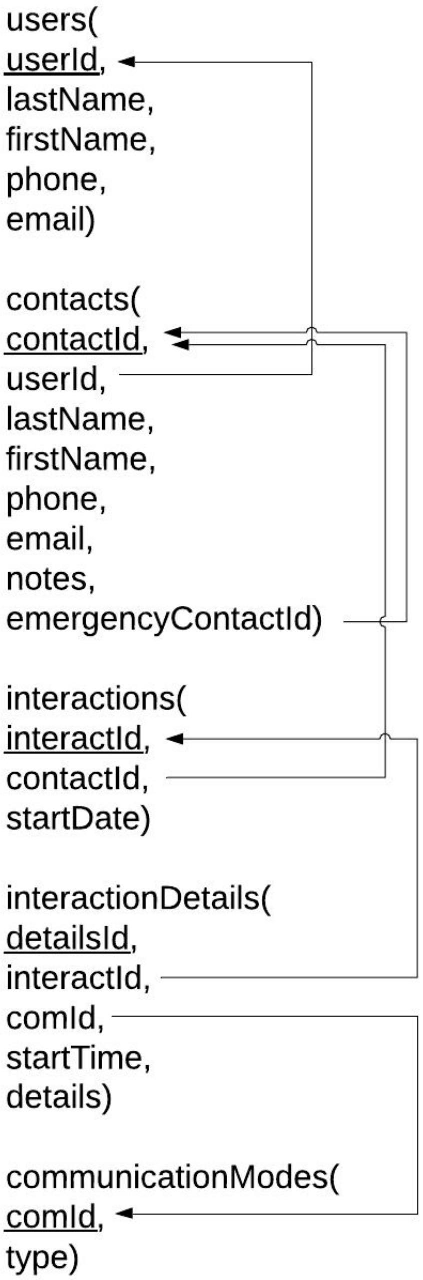
Entity-Relationship Diagram



Note: The contacts table will have an attribute titled emergencyContactsId; it will contain, as a foreign key, the number corresponding to the contactId of another member of the contacts table.

Note: interactionDetails facilitates the M:M relationship between interactions and communicationModes.

Schema



users(
userId,
lastName,
firstName,
phone,
email)

contacts(
contactId,
userId,
lastName,
firstName,
phone,
email,
notes,
emergencyContactId)

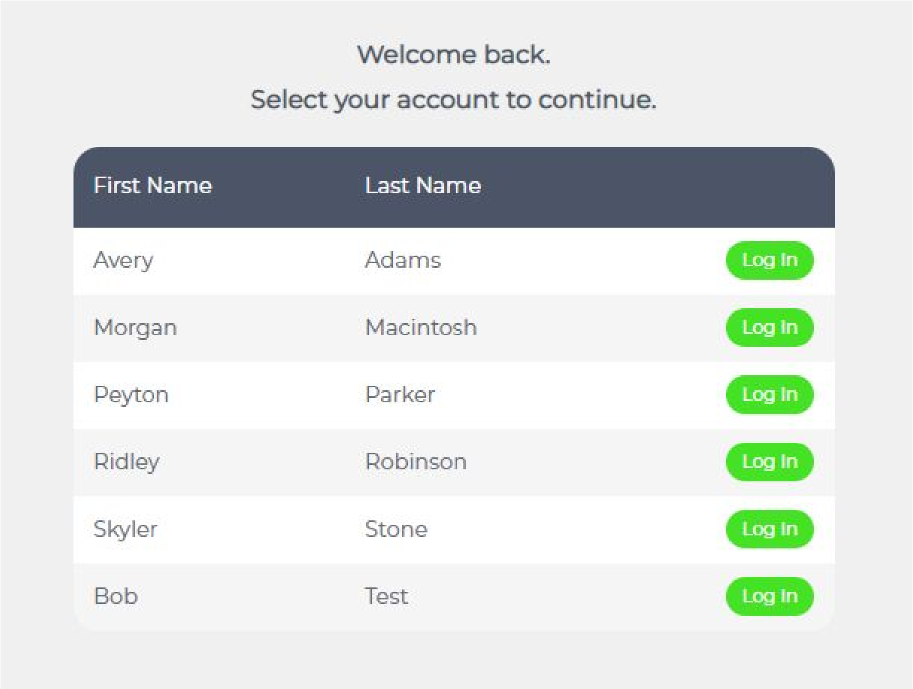
interactions(
interactId,
contactId,
startDate)

interactionDetails(
detailsId,
interactId,
comId,
startTime,
details)

communicationModes(
comId,
type)

Screen Captures

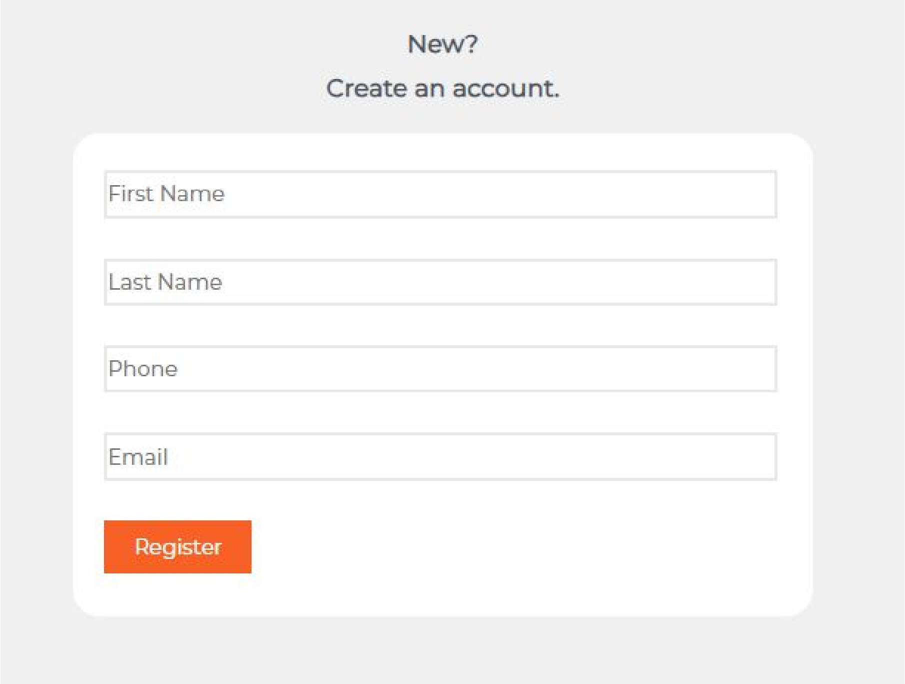
READ/BROWSE/DISPLAY users page section



Welcome back.
Select your account to continue.

First Name	Last Name	
Avery	Adams	Log In
Morgan	Macintosh	Log In
Peyton	Parker	Log In
Ridley	Robinson	Log In
Skyler	Stone	Log In
Bob	Test	Log In

CREATE/INSERT/ADD users page section



New?
Create an account.

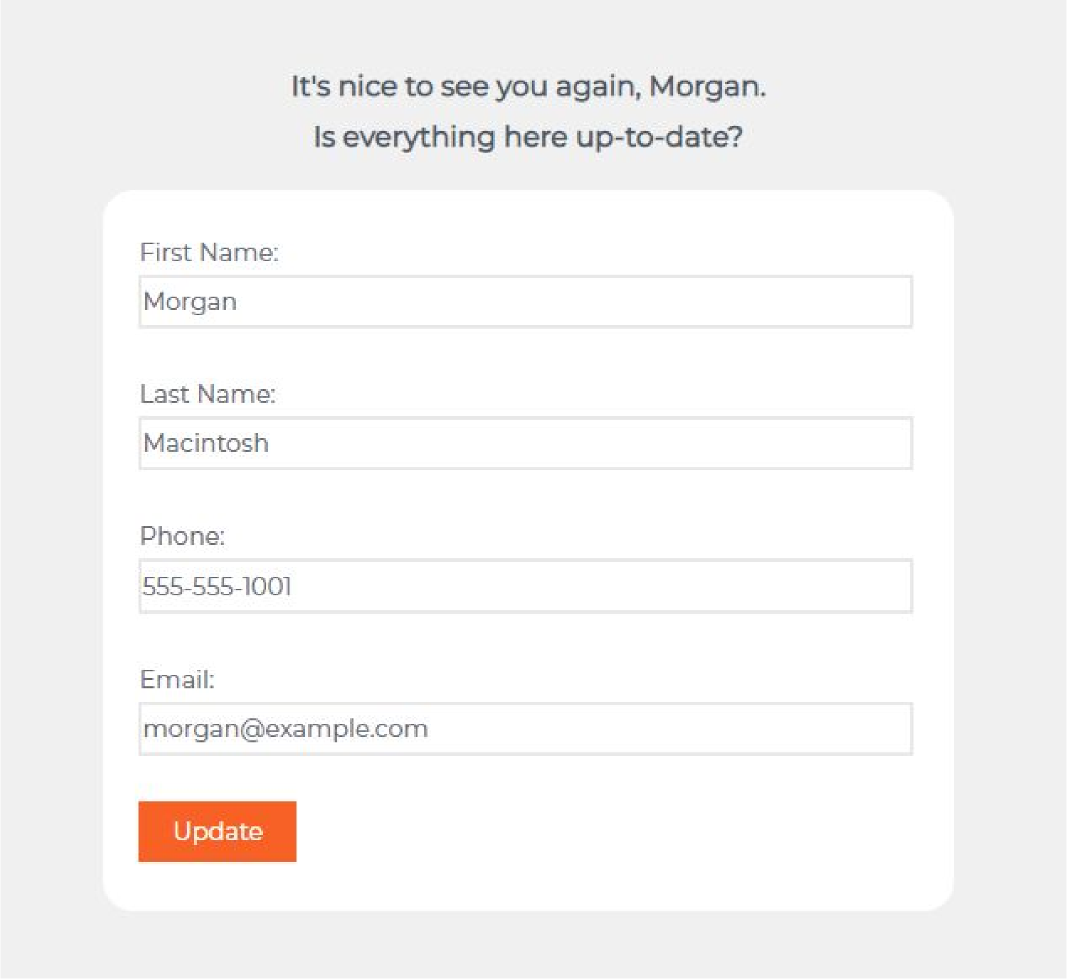
First Name

Last Name

Phone

Email

UPDATE users page



It's nice to see you again, Morgan.
Is everything here up-to-date?

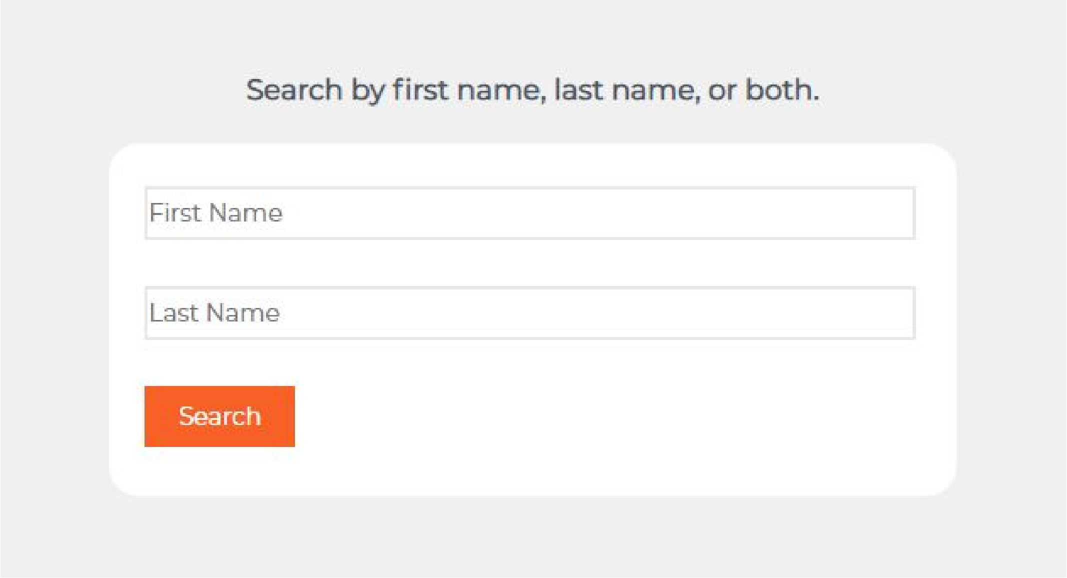
First Name: Morgan

Last Name: Macintosh

Phone: 555-555-1001

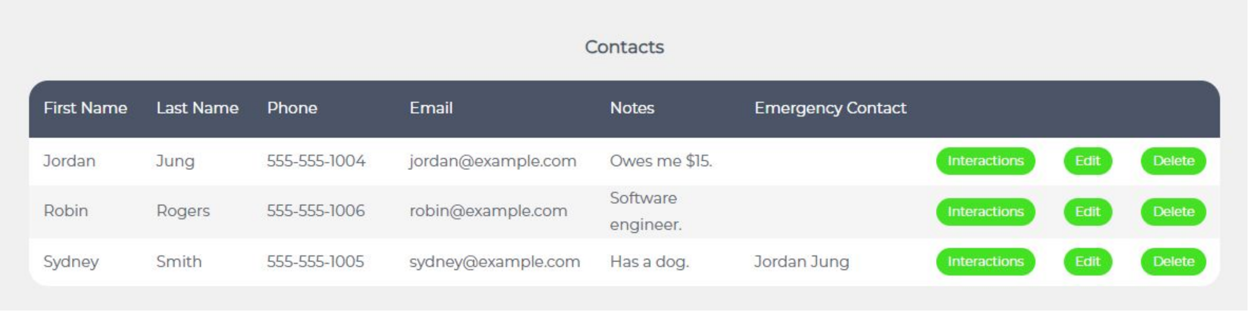
Email: morgan@example.com

SEARCH/FILTER contacts page section



Search by first name, last name, or both.

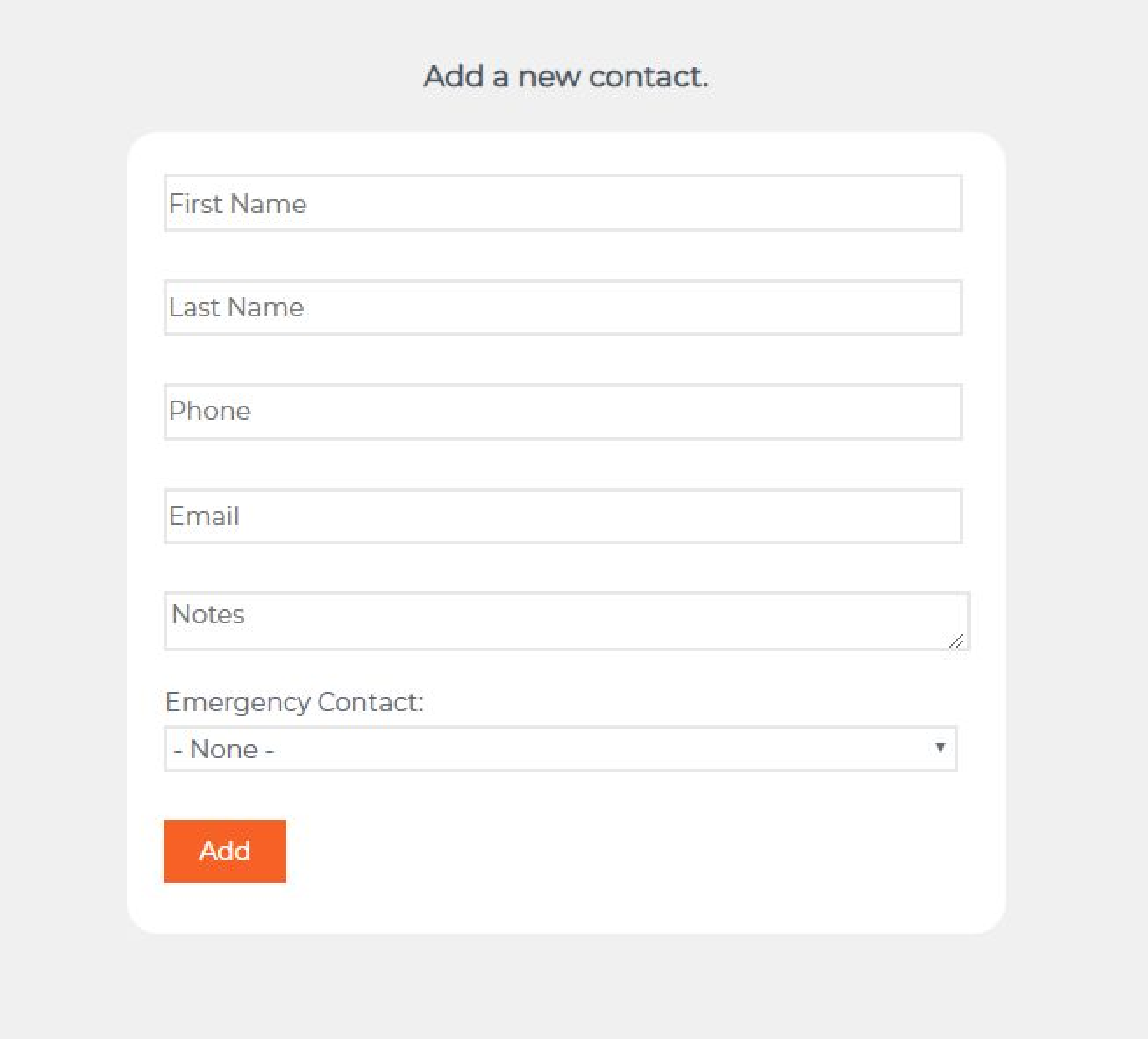
READ/BROWSE/DISPLAY & DELETE contacts page section



Contacts

First Name	Last Name	Phone	Email	Notes	Emergency Contact			
Jordan	Jung	555-555-1004	jordan@example.com	Owes me $15.		Interactions	Edit	Delete
Robin	Rogers	555-555-1006	robin@example.com	Software engineer.		Interactions	Edit	Delete
Sydney	Smith	555-555-1005	sydney@example.com	Has a dog.	Jordan Jung	Interactions	Edit	Delete

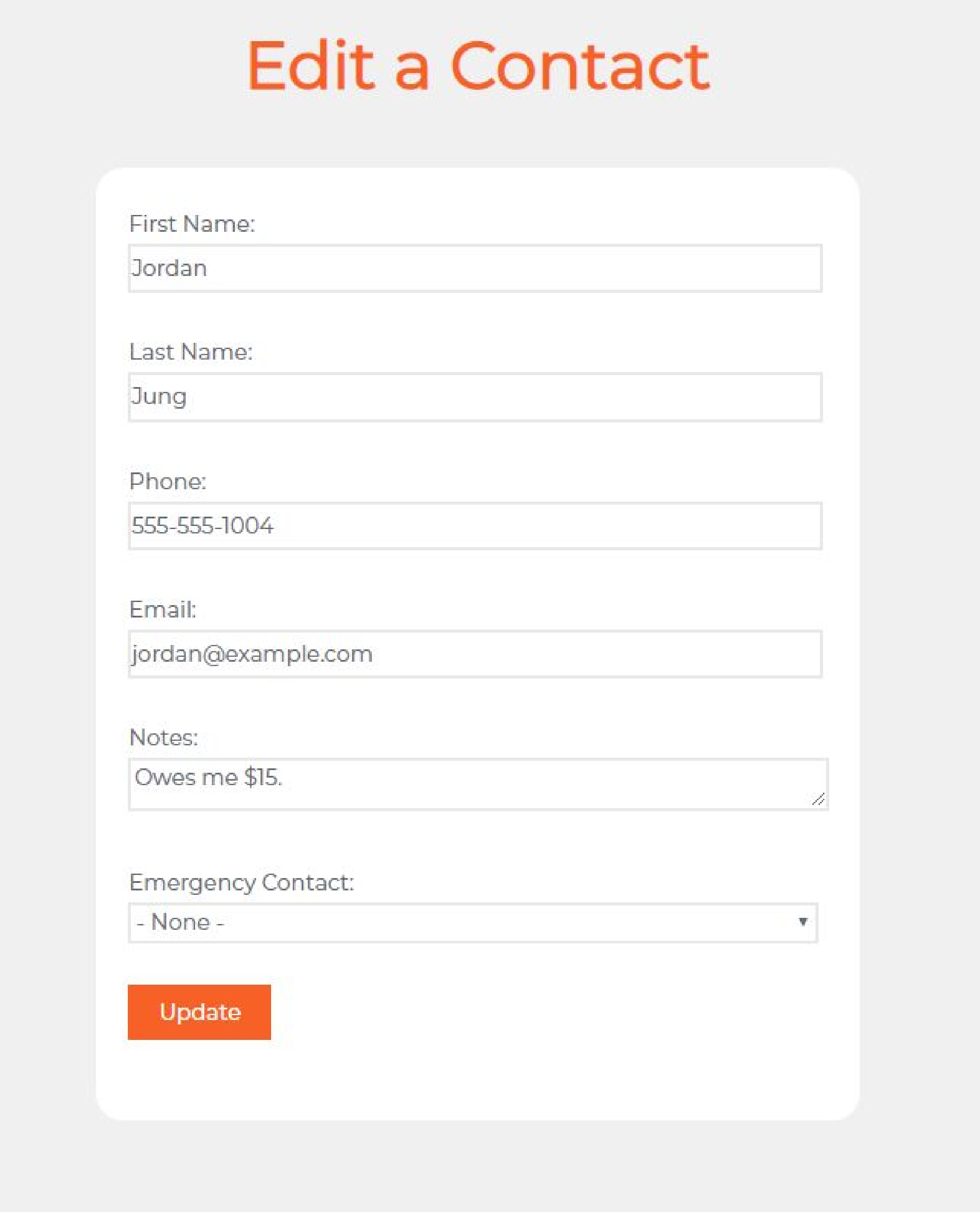
CREATE/INSERT/ADD contacts page section



Add a new contact.

Emergency Contact: - None -

UPDATE contacts page



Edit a Contact

First Name:
Jordan

Last Name:
Jung

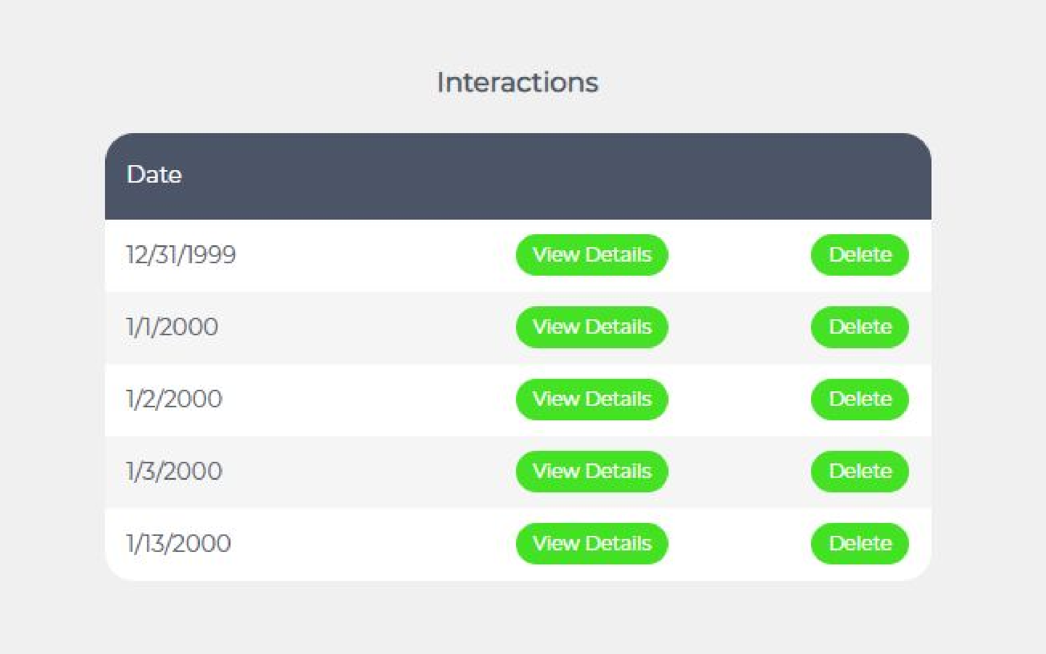
Phone:
555-555-1004

Email:
jordan@example.com

Notes:
Owes me $15.

Emergency Contact:
- None -

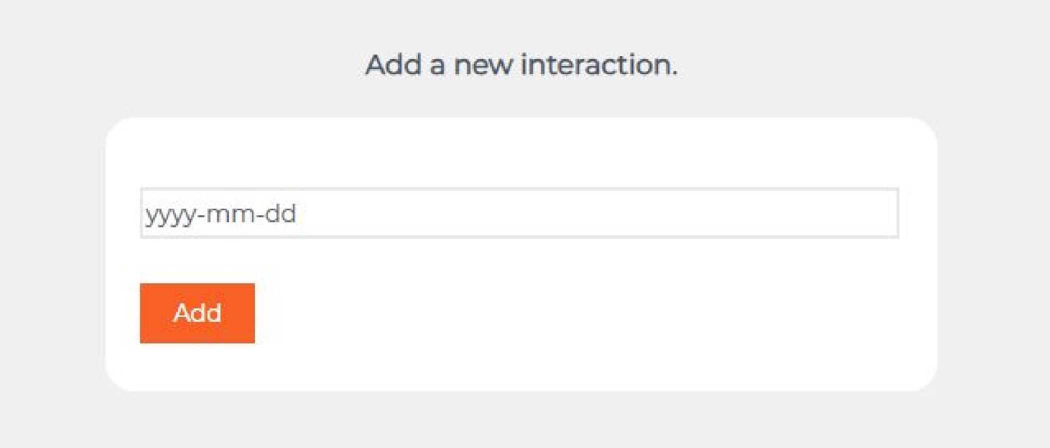
READ/BROWSE/DISPLAY & DELETE interactions page section



Interactions

Date		
12/31/1999	View Details	Delete
1/1/2000	View Details	Delete
1/2/2000	View Details	Delete
1/3/2000	View Details	Delete
1/13/2000	View Details	Delete

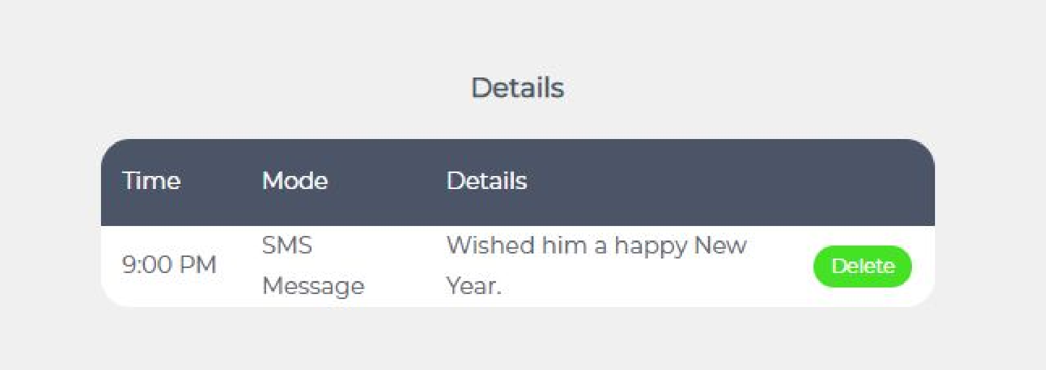
CREATE/INSERT/ADD interactions page section



Add a new interaction.

yyyy-mm-dd

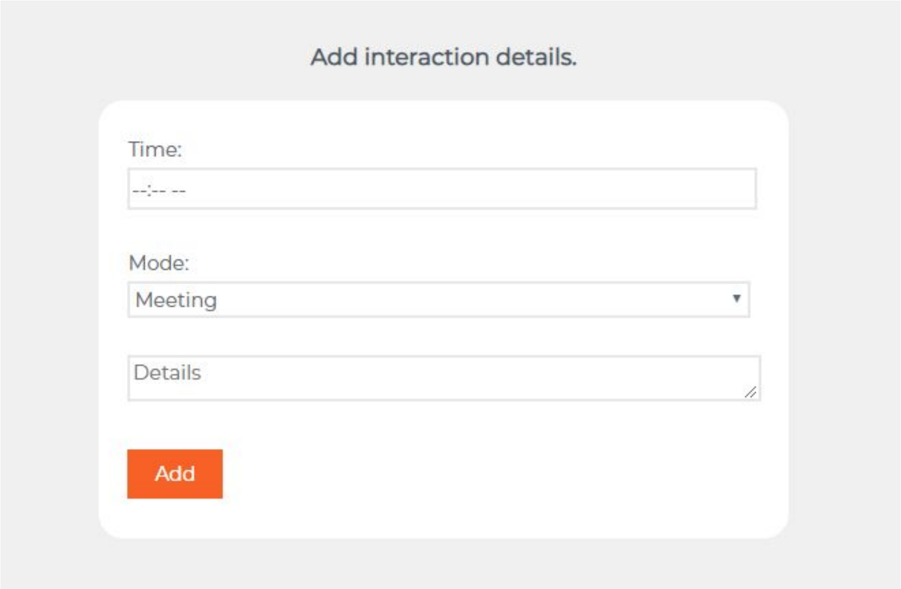
READ/BROWSE/DISPLAY & DELETE interactionDetails page section



Details

Time	Mode	Details	
9:00 PM	SMS Message	Wished him a happy New Year.	Delete

CREATE/INSERT/ADD interactionDetails page section



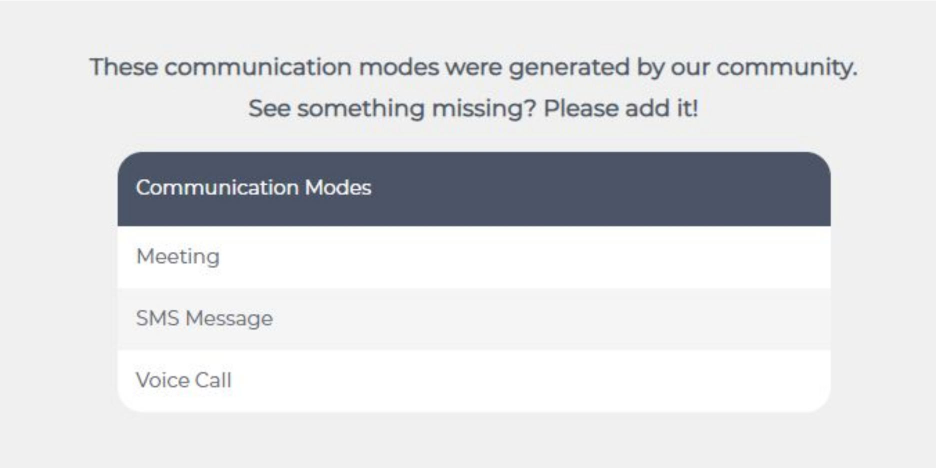
Add interaction details.

Time: --:-- --

Mode: Meeting

Details

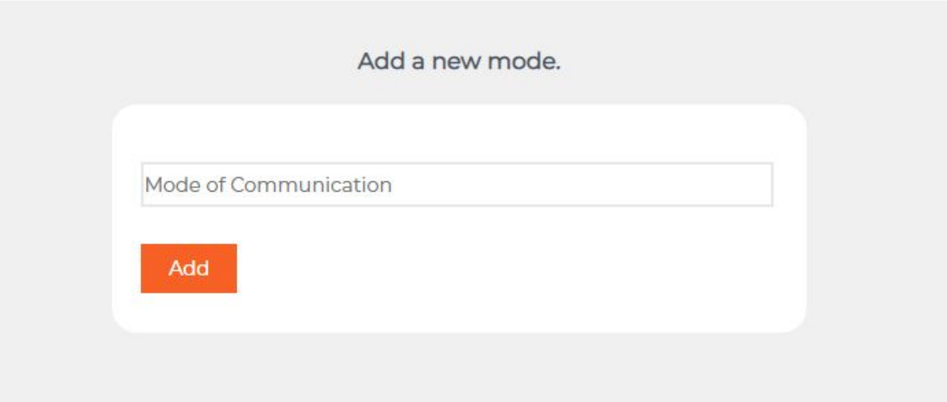
READ/BROWSE/DISPLAY communicationModes page section



These communication modes were generated by our community.
See something missing? Please add it!

Communication Modes
Meeting
SMS Message
Voice Call

CREATE/INSERT/ADD communicationModes page section



Add a new mode.

Mode of Communication