

# Family Interaction Map

Joseph Alflen  
Joshua Inniger  
Ailun Shen

# Introduction

1. What is a Family Interaction Map?
2. Background
3. The Problem
4. Technologies
5. Design Decisions
6. Implementation
7. Testing
8. Security
9. Maintenance
10. Summary

# What is a family interaction map?

- Logical representation of mapping actions, thoughts, and emotions within families to aid in documenting behavior.

A map is divided into two sectors by the relapse line:

- Abstinence sector
- Using-intoxicants sector



Feelings



Thoughts



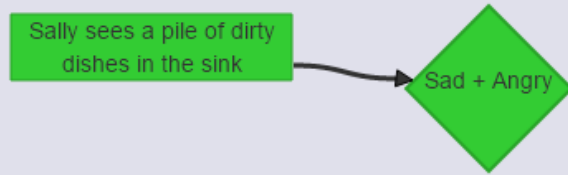
Actions

# Simple Family Map 1

Sally sees a pile of dirty  
dishes in the sink

---

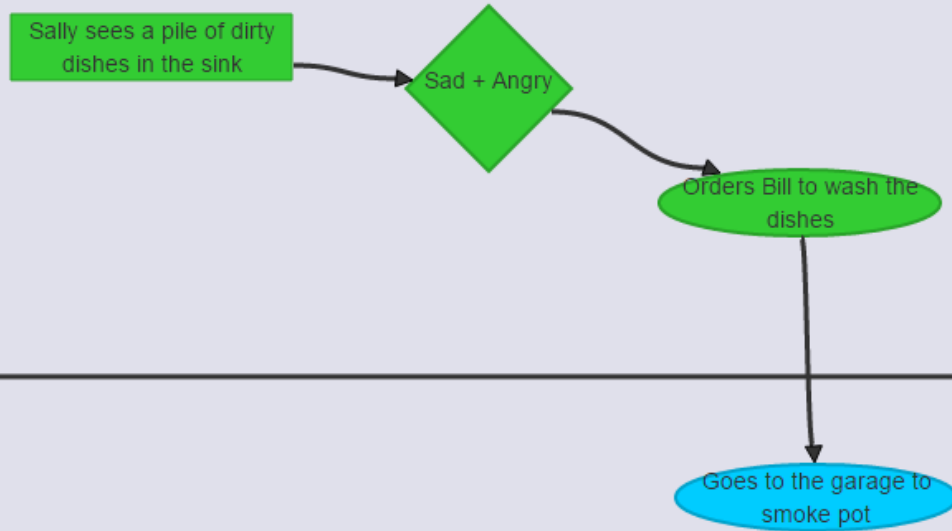
# Simple Family Map 2



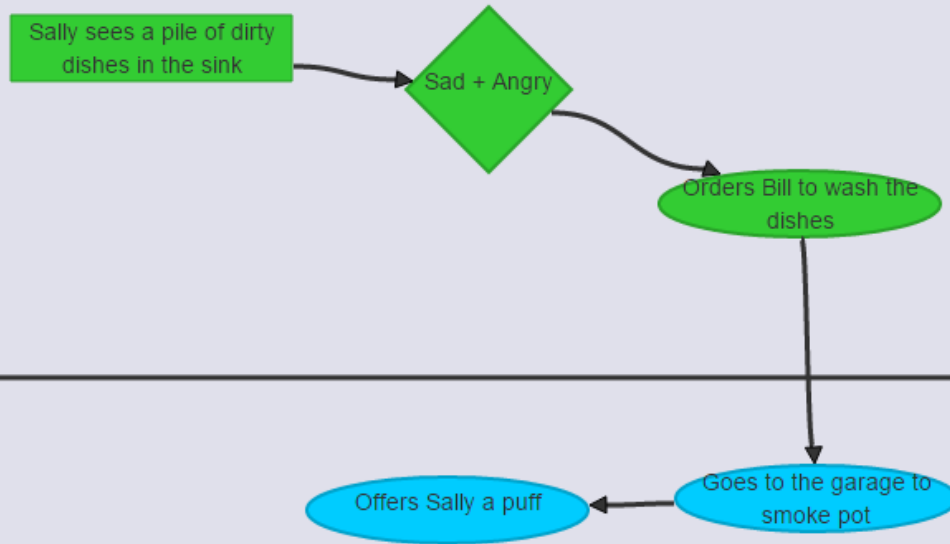
# Simple Family Map 3



# Simple Family Map 4

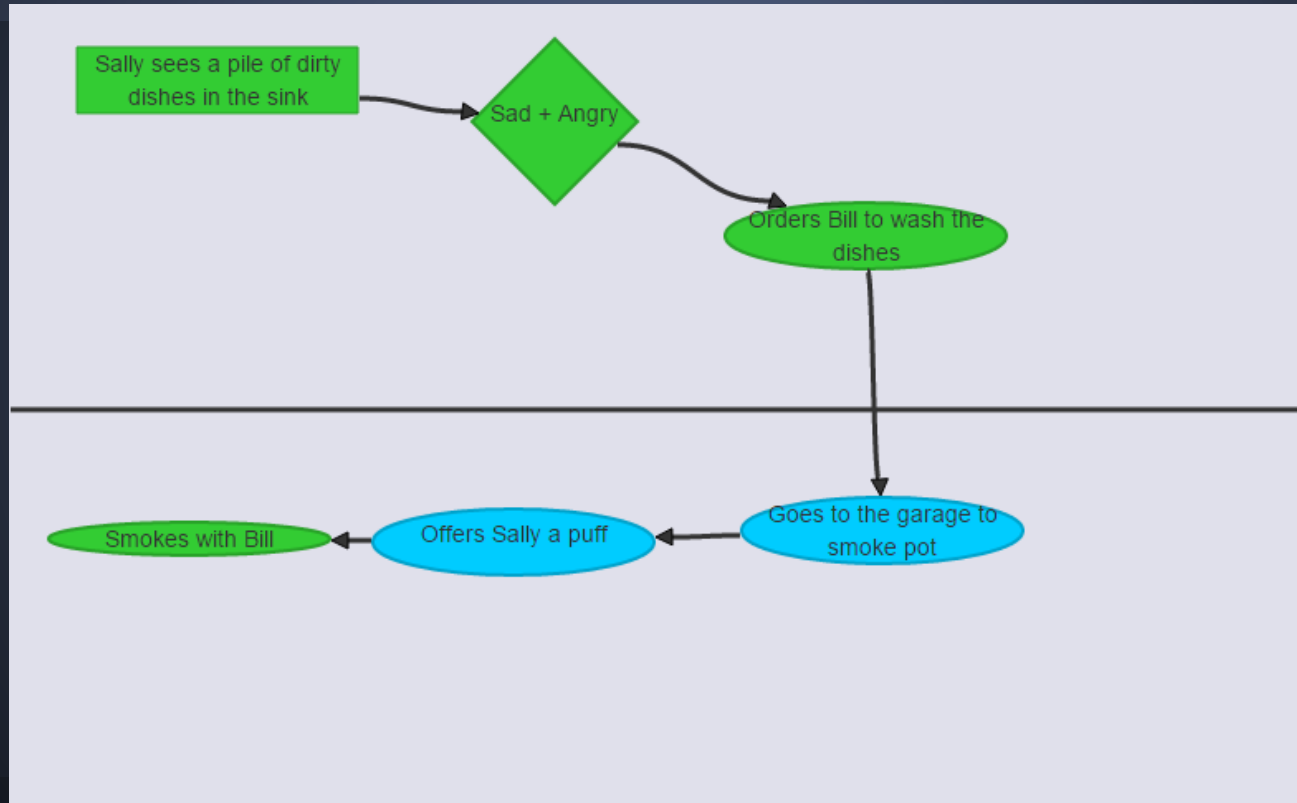


# Simple Family Map 5

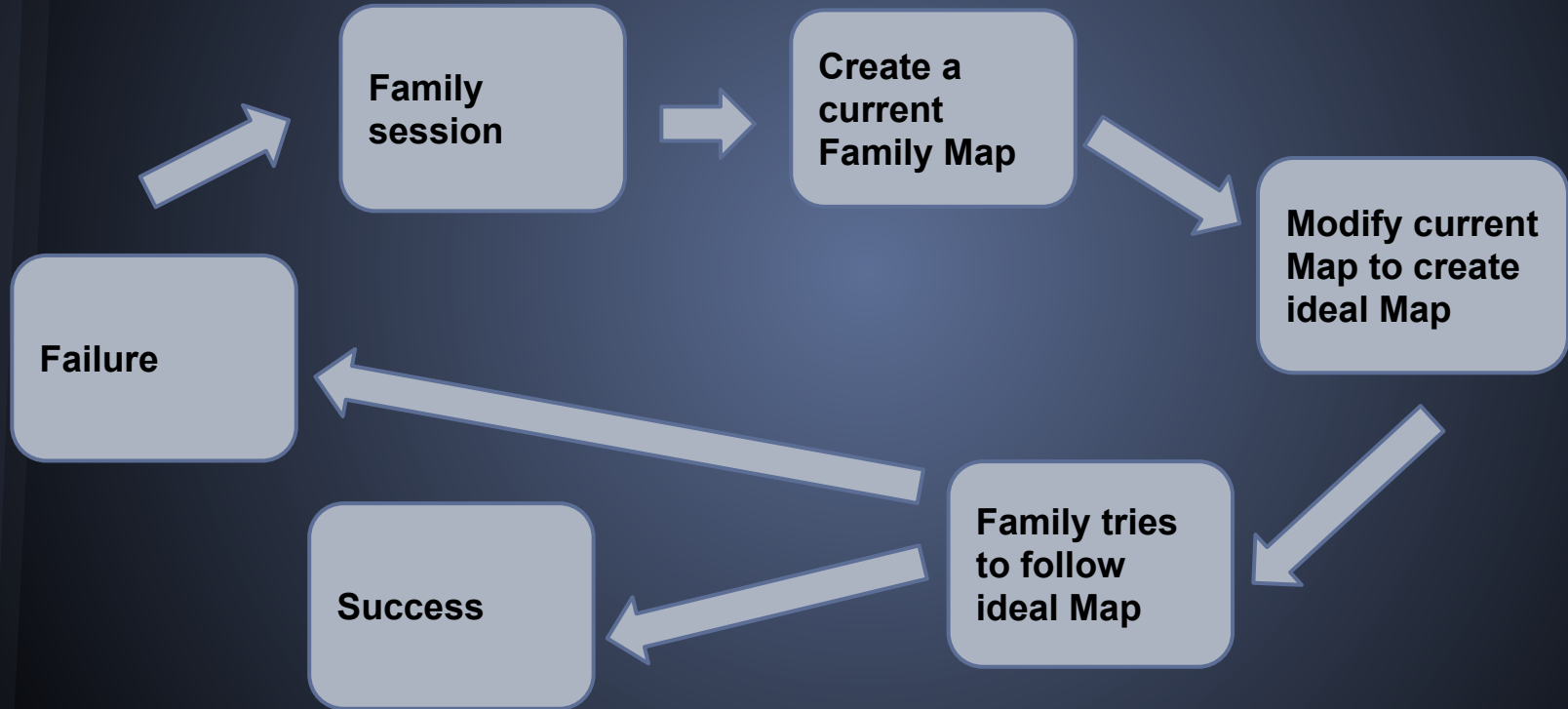




# Simple Family Map 6



# The Process



# Background

This project was taken over and redesigned from previous Senior Design students; William Naylor and Douglas Losey.

What we're covering:

- Client Overview
- Defining the problem
- Technology Used

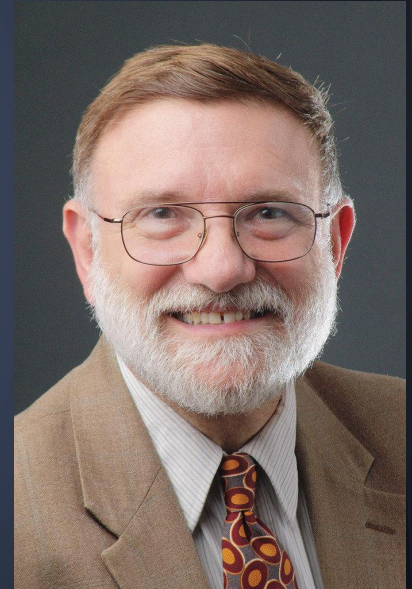
# Client

## Michael R. Liepman, MD, DLFAPA, FASAM

Professor of Psychiatry and Director of Psychiatry Research

Western Michigan University Homer Stryker MD School of  
Medicine

- Raised in Kalamazoo
- Educated at the University of Michigan in Ann Arbor
- Pursued an academic career in Addiction Psychiatry.



# Problem History

- Designed in the 80's
- Used sticky notes and/or whiteboard
- Not portable - Patients couldn't get a copy. Had to be recreated every meeting.

# The Problem

How can we map behavior in such a way that can be successfully interpreted by a professional as well as the patient(s)?

Desired actions:

- Undo/redo
- Save/load
- Print map

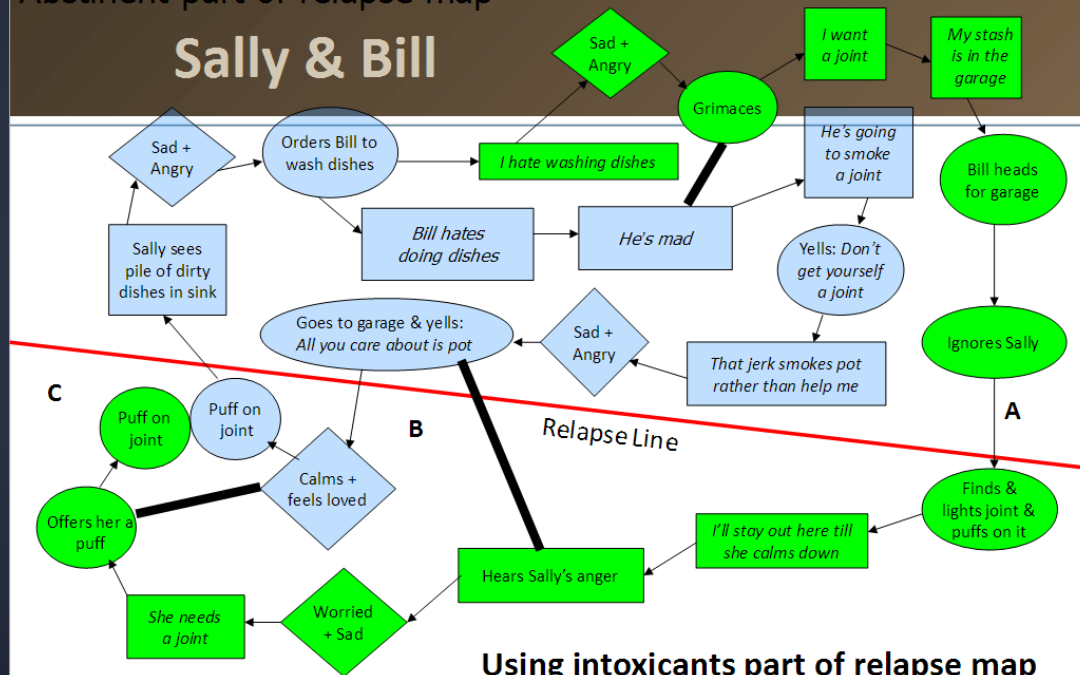
Must be:

- Secure
- Efficient
- Easy to operate

# Family Map Example

Abstinent part of relapse map

## Sally & Bill



Using intoxicants part of relapse map

# Technology

The Family Interaction Map is an interactive web-based application.

## Technologies used:

- HTML/CSS
- Ruby on Rails
- Javascript
- Bootstrap
- JointJS
- MySQL
- Git





# Design Decisions

1. Web Development Framework
2. Javascript Framework
3. Undo/Redo Functionality

# Rails Development Framework

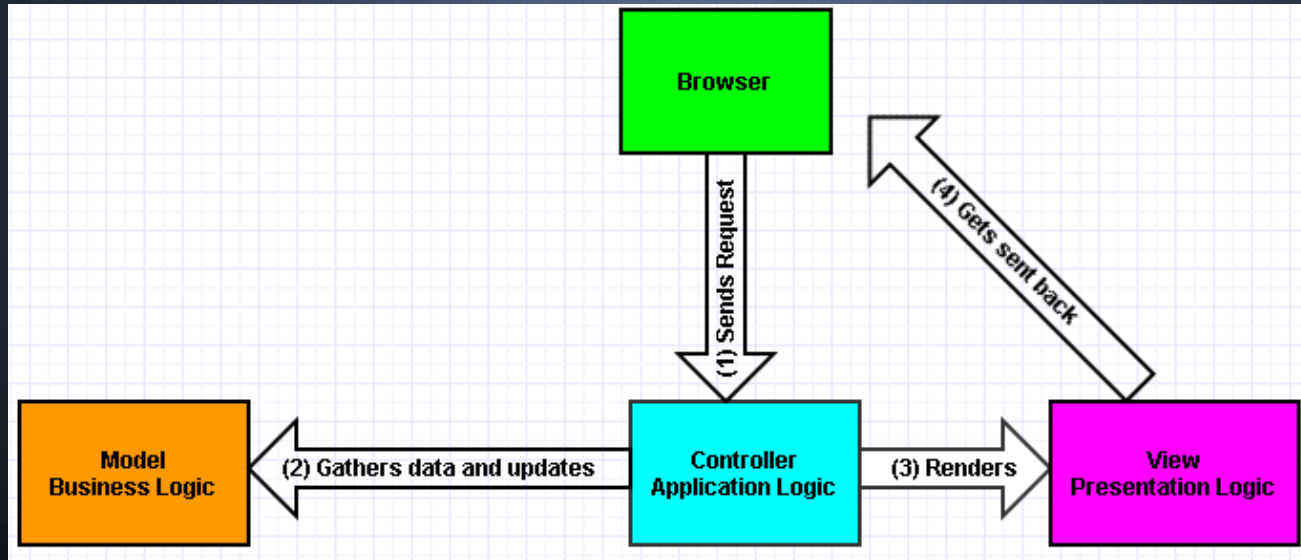
## Why we chose Rails:

- Quick Rapid Development
- Recommended by Dr. Kapenga
- Becoming Industry Standard
- Makes Everything Easier in the long run



# MVC

- Independent logic helps separate the application from the UI.
- Code Reuse(no dependencies on the presentation of data or storage)



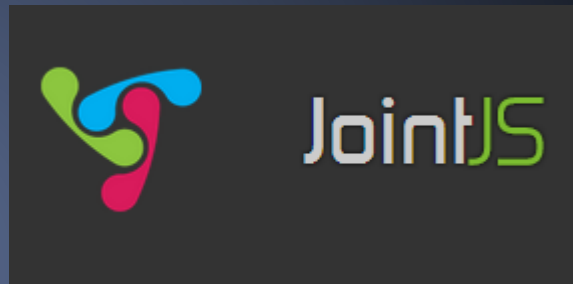
# Javascript Framework

Framework Name	Free to Use	Docs/Examples	Charting Capabilities	Undo
KineticJS	Yes (MIT or GPL Version 2)	Plentiful	None directly built in	No
PaperJS	Yes (MIT License)	Plentiful	None directly built in	No
Joint JS	Yes (Mozilla Public License Version 2.0)	Plentiful	Many directly built in	Yes

**Ultimate Decision - JointJS**

# JavaScript Framework Continued

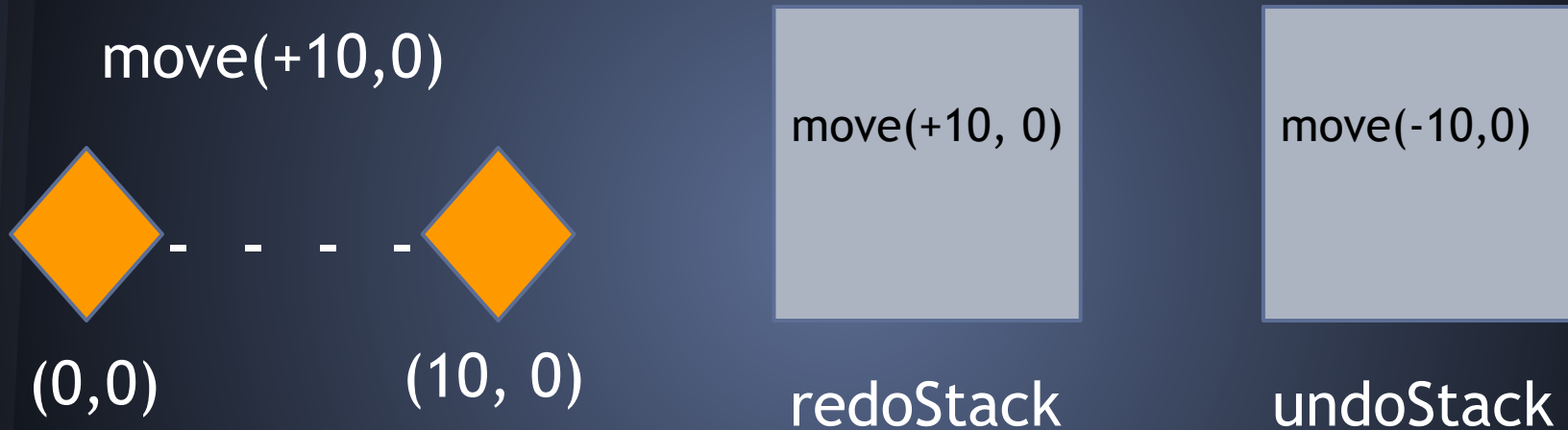
- Serialization of map data to JSON
- Many specific charting examples
- Easy to use and intuitive
- Built in Undo/Redo Functionality



```
toJSON    graph.toJSON()
```

```
fromJSON  graph.fromJSON(json, [options])
```

# Undo/Redo Functionality



JointJS features a Command Manager that listens for graph changes and stores the opposite command.

# Design

Careful examination of previous design along with direct client input lead to the core design.

Four important parts considered:

1. Browser Side
2. Server Side
3. Compatibility
4. Encryption

# Browser Side

## Javascript

- JointJS provides a canvas and tools for creating and manipulating shapes.
- Allows for easy storage of the map via JSON.

## Web Design

- Bootstrap Framework neatly organizes the HTML, CSS, and Javascript.
- Features a simple yet elegant style.

Bootstrap

JointJS

JavaScript



# Server Side

Family Map

Ruby

Rails

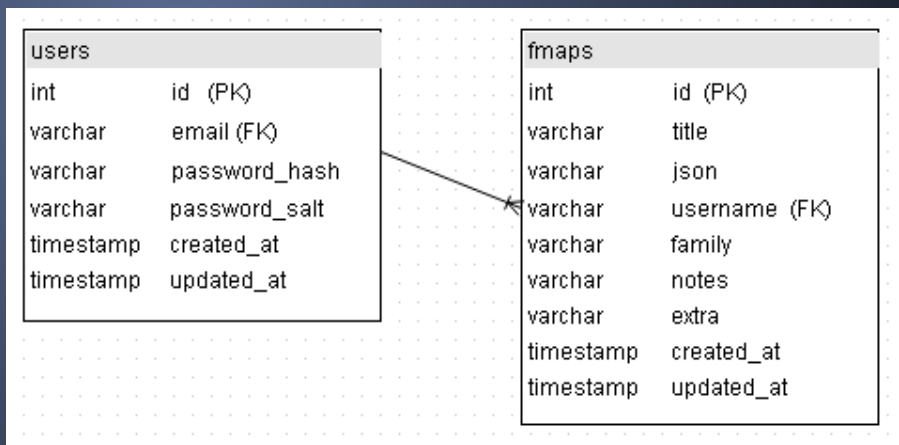
Apache

# Server Side

## Ruby on Rails

- Seamlessly handles the transactions with MySQL

## MySQL



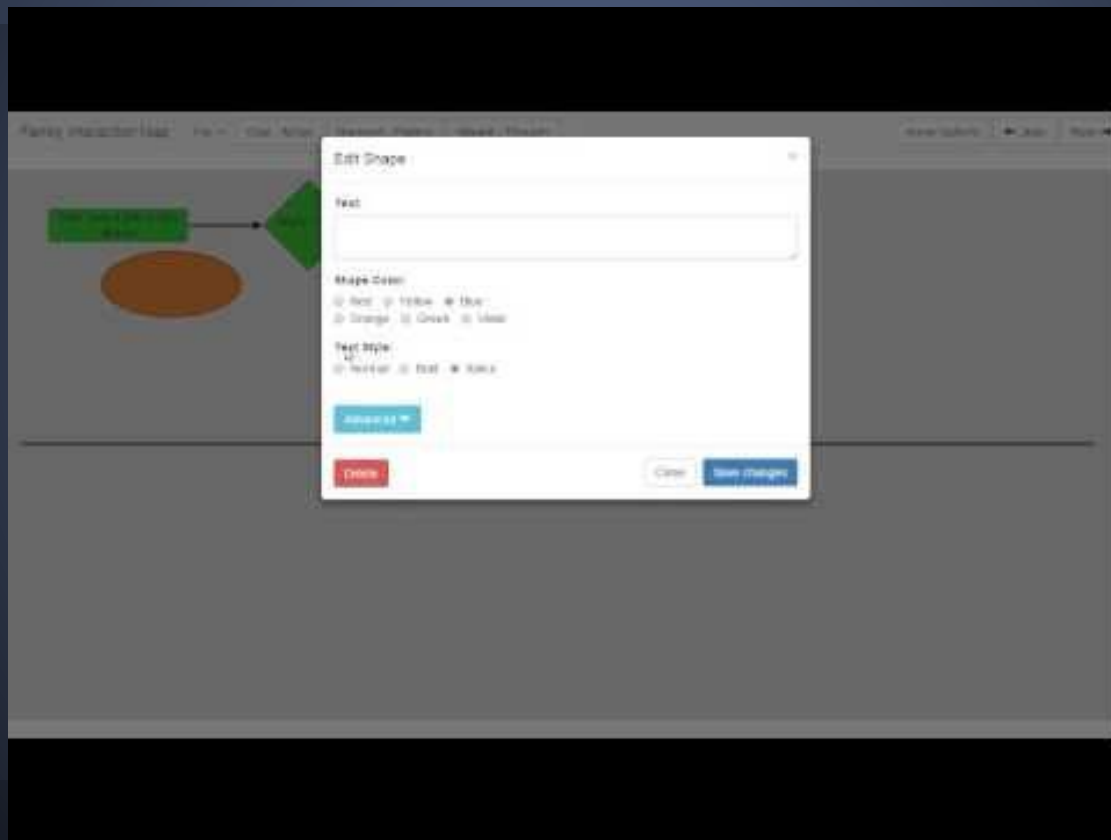
# Compatibility

The Family Interaction Map is designed to be compatible with most modern browsers.

- Google Chrome
- Mozilla Firefox
- Opera
- Safari



# Implementation



# Testing

Exhaustive testing has been done to ensure performance, security, and client satisfaction.

Areas of main focus include:

- Rails database transactions
- Javascript saving and loading
- JSON Decryption

# Rails Testing

Rails features a safe testing environment that will keep all changes from affecting the development and production environments.

By default, every Rails application has three environments: development, test, and production. The testing environment comes in handy in this case.

Rails produces skeleton test code as it is developed.

- Fixtures for sample Family Maps.
- Models tests

```
$ bin/rake test test/models/article_test.rb  
test_should_not_save_article_without_title  
.
```

```
Finished tests in 0.047721s, 20.9551 tests/s, 20.9551 assertions/s.
```

```
1 tests, 1 assertions, 0 failures, 0 errors, 0 skips
```

# Javascript Testing

QUnit Javascript Framework v1.17.1

Supports IE6+ and current -1 for Chrome, Firefox, Safari and Opera.

## QUnit Example

☐ Hide passed tests ☐ Check for Globals ☐ No try-catch

Filter:

Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)  
Chrome/40.0.2214.115 Safari/537.36

Tests completed in 7 milliseconds.  
1 assertions of 1 passed, 0 failed.

1. **hello test (1)** Rerun 3 ms

# JSON Decryption Testing

## Procedure:

1. Create duplicate “dummy” Family Maps
2. Encrypt and then Decrypt the maps
3. Compare to check for differences



# Security

Due to the health care environment in which this system will be used, security is top priority. Patient information must be kept confidential.

The primary security measures are:

- User authentication
- Session encryption
- Data encryption

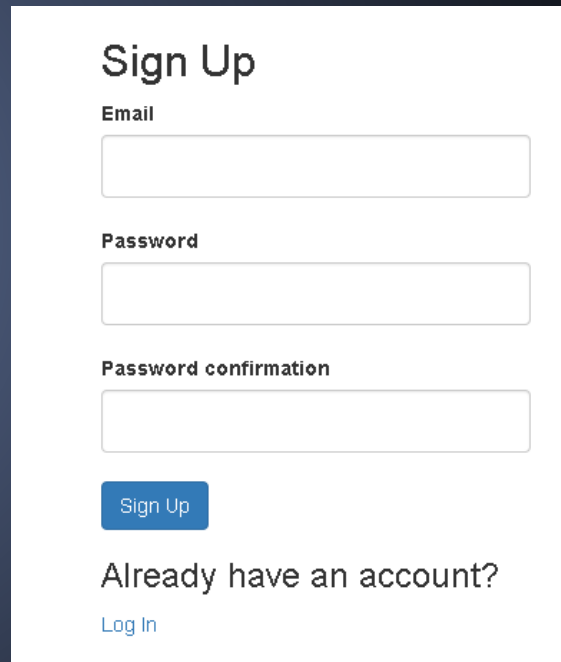
# Security Issues and Rails Solutions

- Session Hijacking
  - Rails allows for easy implementation of SSL(Secure Socket Layer)
- Cross Site Request Forgery
  - Rails protects from URL manipulation
- Malicious Code in File Uploads/Downloads
  - File sanitization removes malicious code
- SQL Injection
  - Automatic rejection of harmful commands

# User Authentication

The user signs up and signs in with an email and password of his/her choice.

User credentials are encrypted with BCrypt Ruby Gem and then stored in the database.

A sign-up form with a white background and rounded corners. It contains three input fields for email, password, and password confirmation, each with a label above it. A blue 'Sign Up' button is positioned below the third field. At the bottom, there is a link 'Log In' and a text prompt 'Already have an account?'.

**Sign Up**

Email

Password

Password confirmation

[Sign Up](#)

Already have an account?

[Log In](#)

# Session and Data Encryption

## Session Encryption

SSL (Secure Sockets Layer) encrypts the connection.

## Data Encryption

256-bit Advanced Encryption Standard (AES) protects the metadata associated with the created diagram. Crypto-JS Javascript framework was used.

# HIPAA Compliance

By collecting and storing PHI(Protected Health Information), we must remain HIPAA (Health Insurance Portability and Accountability Act) compliant.

The features we implemented to be HIPAA compliant:

- Unique User Identification
- Automatic Logoff
- Encryption and Decryption

- **Encryption and Decryption - 164.312(a)(2)(iv):** Implement a method to encrypt and decrypt electronic protected health information.
- **Encryption - 164.312(e)(2)(ii):** Implement a mechanism to encrypt electronic protected health information whenever deemed appropriate.

# Maintenance

Due to our testing strategy and regular client interactions, maintenance is very minimal in the foreseeable future.

- Full maintenance documentation will be provided future developers.
- Includes full installation instructions

# Summary

## Remember...

- A Family Interaction Map is used for behavioral analysis.
- Client desired a secure and efficient solution.
- We designed and tested a Rails powered web app suitable to our clients needs.

# Questions?



**Thank You!**