InfoBox
A Personal Information Manager
Built with Ruby on Rails

Dustin Martin
What is a personal information manager?
A personal information manager is a piece of software that functions as a personal organizer. It allows for the storage of various forms of information such as to-do lists, notes, dates, etc.
InfoBox is an online personal information manager. It allows for the storage of notes, photos, links, lists, and files, all of which can be organized with tags.
Tags: A method of organizing data by assigning a **keyword** to the data. Each piece of data can be assigned multiple tags and each tag can be assigned to multiple pieces of data. Tags allow for much more flexibility than hierarchical organization methods.
For example, the to-do list item “get car oil changed” could be tagged with car and maintenance. If someone looked at all data associated with the tag maintenance, all items with that tag, including “get car oil changed” would be displayed.
What is Ruby on Rails?
Ruby on Rails is a combination of two things. Ruby, a programming language that originated in the mid 1990’s, and Rails, a programming framework that makes creating web applications with Ruby far easier.
Ruby on Rails applications are developed using the Model-View-Controller (MVC) architectural pattern.
The MVC architectural pattern is a method of organizing the code. It allows one to separate data (model) and user interface (view) concerns.
This means changes to the user interface do not affect the data handling, and that the data can be reorganized without changing the user interface. This is accomplished through the use of an intermediate component: the controller.
<% for note in @notes %>
  <tr>
    <td>
      <h4><%=h note.send("title")%></h4>
      <p><%=h note.send("contents")%></p>
      <p><% if note.tags.size.nonzero? %>
        <div class="tags">
          <strong>Tags: </strong><%= render :partial => 'tag', :collection => note.tags %>
        </div>
        <% end %>
      </p>
      <p>
        <%= link_to 'Show', :action => 'show', :id => note %> -
        <%= link_to 'Edit', :action => 'edit', :id => note %> -
        <%= link_to 'Destroy', { :action => 'destroy', :id => note.id }, :confirm => 'Are you sure?', :method => :post %>
      </p>
    </td>
  </tr>
<% end %>
Model Code - note.rb

class Note < ActiveRecord::Base
  acts_as_taggable
  belongs_to :user
end
class NotesController < ApplicationController
  before_filter :login_required

  def list
    @note_pages, @notes = paginate :notes, :per_page => 10,
      :conditions => ['user_id=?', current_user.id]
  end

  def create
    @note = Note.new(params[:note])
    @note.user = @current_user
    if @note.save
      flash[:notice] = 'Note was successfully created.'
      redirect_to :action => 'list'
      @note.tag_with params[:tags] if params[:tags]
    else
      render :action => 'new'
    end
  end
end
What were some of my goals for this project?
1.) Create a place where people could store **multiple forms of information**. Currently InfoBox supports the storing of **notes, photos, links, lists and files**.
2.) Allow for **easy organization** through the use of **tags**.
3.) Allow for multiple users of InfoBox and **protect** each user’s **stored information** through the use of passwords.
Now lets take a look at InfoBox
Problems I had while working on this project.
1.) **Understanding how Rails interacts with the database.** Rather than writing raw SQL, Rails uses an intermediary layer to access the database. This allows for a database-neutral applications but is confusing when coming from other languages that do not take this approach.

**Ruby on Rails Code**
```
partname = "gearbox"
b = Part.find(:first, :conditions => [ "name = ?", partname ])
```

**SQL Code**
```
SELECT * FROM parts WHERE name = 'gearbox' LIMIT 1;
```
2.) **Insufficient planning** on the front end on what features *exactly* I wanted to include and which ones should be done first. The limited planning I did lead me to develop in a very unorganized fashion and caused me to bounce around between developing features which then **slowed me down**.
Future plans for the project.
1.) The **addition of other sections** for the storage of other information such as **contacts or a calendar**.
2.) A **revamped user interface** that provides easier and quicker access to all the user’s information.
3.) Allow for public content so that a user can share information with others, even if they do not have an account.
3.) **Other plans include:**

- Stronger security
- Validation for entered information
- Image processing for thumbnail generation
Questions?