



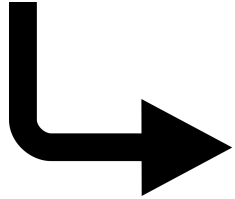
Plant-Clinique

Osamah, Roman, Prabu,
& Eric



What is *Plant-Clinique*?

- Allows management and visualization of user plants
- Facilitates the flow of plants information
- Allows extensive plantcare questions about your plants



Ultimate goal: To aid users with plant care and save plants from uninformed mistreatment

Problems Addressed

- People lacking plantcare knowledge and forgetful about taking care of their plants
- The inaccessibility to

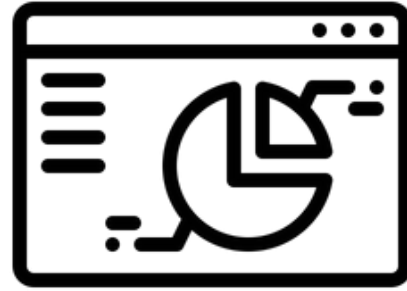


Features

User Authentication

Clearance gem

- Secured access to own accounts
- Forgot Password
- Remember sign in

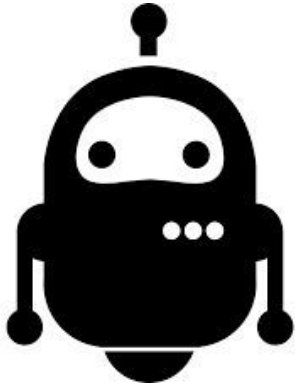


User Dashboard

- Observe all user's plants information at a glance
- Modern display

Chatbot

- Basic conversational commands
- Never lost messages
- Command options to simplify and speed up conversation
- Uses Trefle.io (plant API) for plant information based on user specified plant type



Forum: The Post Pages

- List of topics to quickly filter posts
- Intuitive design
- Related posts on users page

Reminders

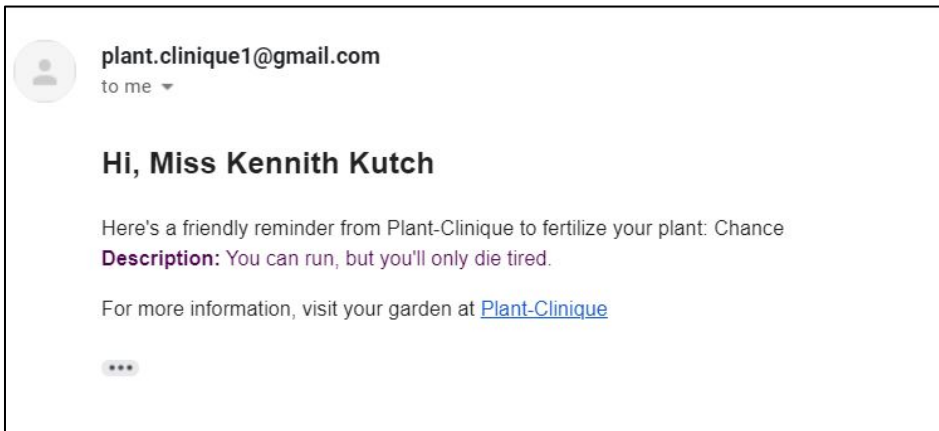
Lets users create a reminder for their plants

- Reminder time
- Reminder's description

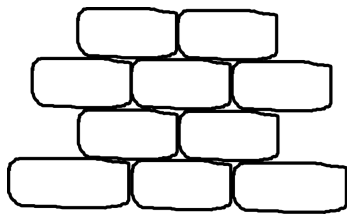
Using Heroku Scheduler to email reminders, making sure no plant is forgotten about

Notifications

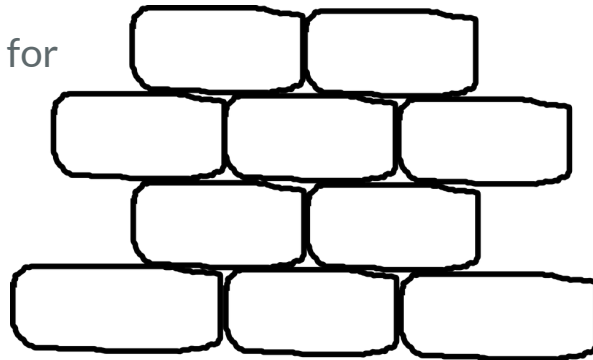
- Notifies users of comments on their posts
- Both have a badge visible in navbar

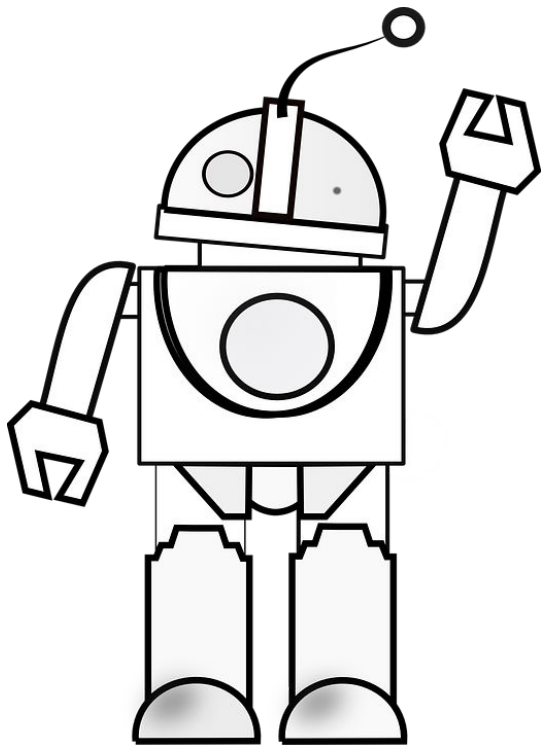


Challenges



- Javascript problems regarding how reactive the navbar could be (solved)
- The footer sticking to the bottom of all pages
- A crucial API (Trefle API) terminated early May (RIP)
- Navigating Gmail authentication for ActionMailer (solved)
- Managing time between making new features and polishing up existing features
- Using CableReady and Redis did not work in Heroku for us
- GitHub actions worked for rubocop but remained buggy for tests when the mimemagic library got yanked





Other Cool Technology

Using Trefle, OpenWeather, and Ambee Soil API to get the necessary plant care information (plant info, weather, soil)

Generally clean UI design with reactive webpage (Collapsible navbar, reactive resizing, reactive badges)

Staging and production repo (continuous integration)

Gems for better debugging and clean codes: `Binding_of_caller`, `rubocop`, `better_errors`, `awesome_print`

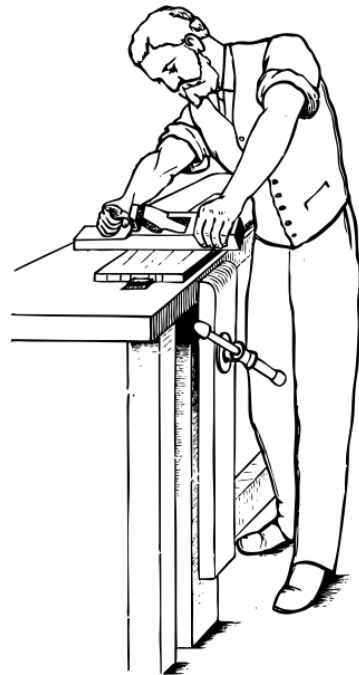
Testing

We implemented automated testing

Using Rails-controller-testing gem for Rails MVP
integration test

Using RSpec: Integration testing for Clearance and
RailsAdmin gems

- User authentication
- Security



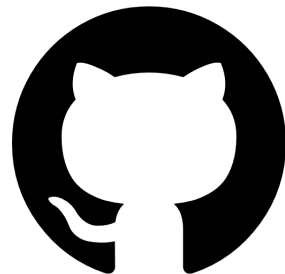


How we Collaborated

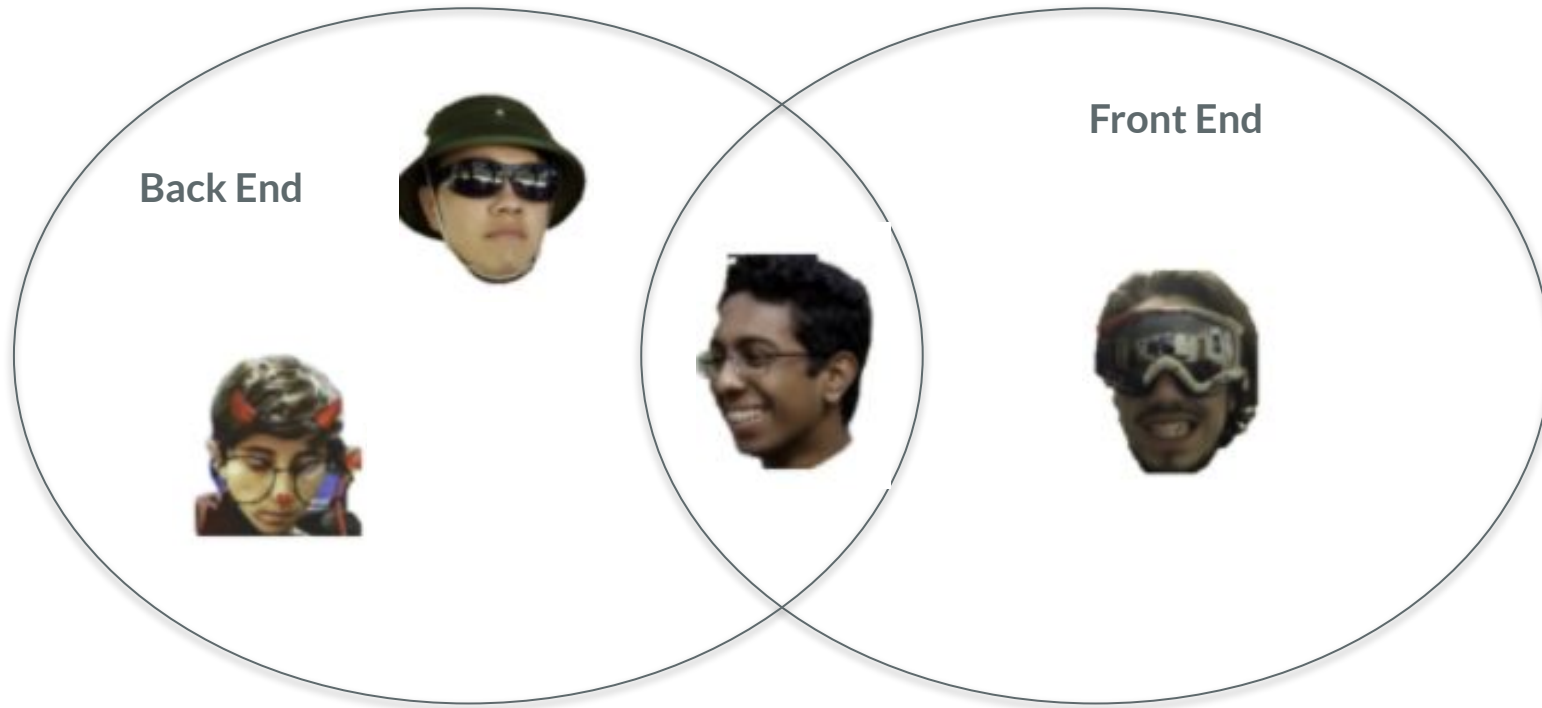
Our team from the get go have been utilizing all tools available for communication and collaboration.

Tools we've been using

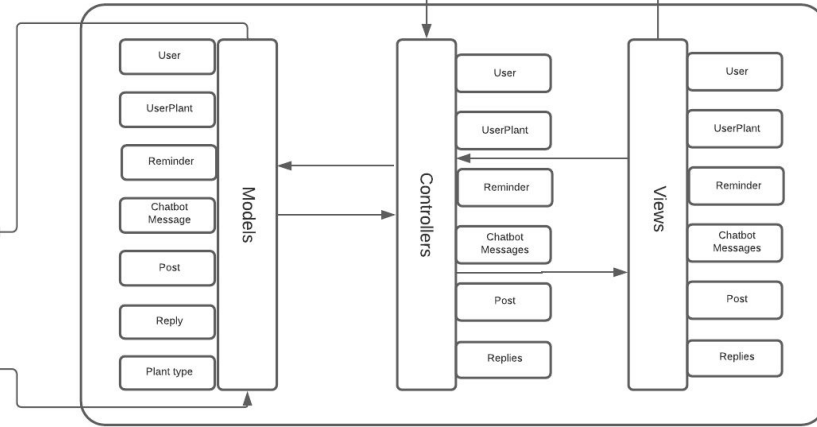
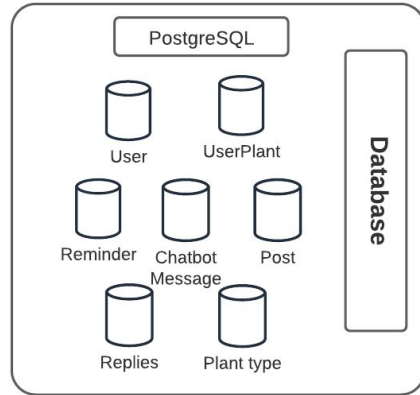
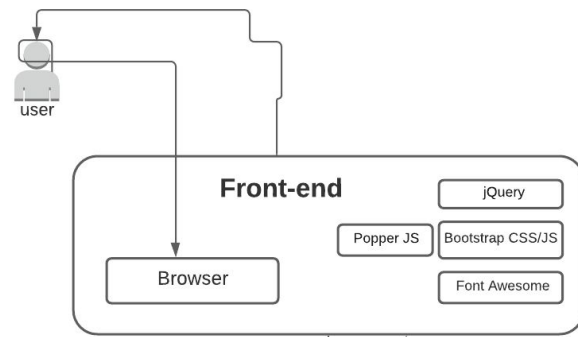
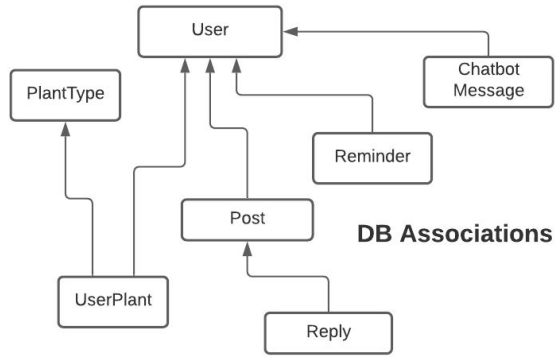
- Communications: Slack, Zoom, Topia, Meet
- Project managements: Trello
- Code/file share Github, Google Drive
- App hosting: Heroku
- Interface, design: Firma, Lucid Chart



Division of Development




Architecture



Our App Demo

Team Reflection: What we're proud of

- A working application so that anyone and everyone can take care of their plants
- Email Notifications
- A beautiful UI
- The many API we have used :
 - Trefle.io 
 - Mapbox
 - Pixabay
- The chatbot steps which allow for more detailed feedback

Team Reflection: Where we would go from here

- We would have loved to bolster the chatbot with
 - A feature that would survey what you were looking for and predict what plant would be perfect
 - A feature that would use APIs to predict what plant would be easiest to grow where you are
- An up-to-date weather warning system on the maps
- Real-time in-app notifications
- Ability to take photos and directly upload
- A like and dislike in the posts feature
- Use AJAX to speed up replying and posting
- Track users' milestones
- Turn app into a PWA

Team Reflection: Work Division Strategy

What we would do per week was very much a choice system, with a full trello we just made sure we all had comparable workloads and encourage people to take more if they were light on work. Our due dates were saturday check ins. **Interest Based.**

General:

Backend:

- Osamah: Maps, Notifications, Posts, Chatbot, Login
- Prabu: Gravatar, Script
- Roman: Reminders, Login

Front End:

- At the start, Prabu
- Eric took any front end work he could get

Lessons Learned

- Murphy's Law: "Anything that can go wrong will go wrong"
- With everyone taking a full class load and all virtual meetups, teamwork without communication is actually nothing - check ins were the lifeblood of progress

Brighter Side:

- Trello, Slack, and the many other ways we communicated with each other were essential and being big on communication is the only way to get anything done
- Proper coding practices and professionalism with big projects

Thank you!