TALTECH

School of Information Technologies

Stefan Joel Kivvistik 206178IADB

WEB APP FOR TRACKING PLANT CARE

Project in Building Distributed Systems

Supervisor: Andres Käver

Tallinn 2023

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Stefan Joel Kivvistik

20.02.2023

Table of contents

1 Introduction

There is a lack of free and high-quality options for those looking to manage their gardening duties. The author has great personal interest in growing houseplants and is looking to provide an open-source alternative to some of the other tools that are available on the market as of February 2023.

The goal of this project is to create a web app that would assist users with keeping track of watering, repotting and fertilizing their plants. In addition to setting reminders, the user should be able to track any pests and diseases their plants may have, organize their plants with the help of collections or tags and keep a history of anything that has happened to any specific plant.

This project is a part of coursework for the course Building Distributed Systems in TalTech university and will therefore comply with any demands and limitations set by the lecturer. ERD schema to be used for this project is included in the appendix of this document.

2 Scope of work

Critical features:

- User can add plants.
- User can move a plant to a collection and create new collections.
- User can make tags and assign them to plants.
- User can set up reminders for each plant. Base reminder types available are watering, fertilizing, pest control and repotting, but user can make custom reminders as well.
- User can create history log entries for any specific plant.
- User can mark pests or diseases for a plant and manually track their severity.
- User can add photos for their plants.

Secondary features:

- User can mass create history entries for several plants at once (for example, create entry for every plant in a collection).
- User can get reminders sent to email.
- User can request scientific info about a specific plant through the webapp. This could be implemented with Trefle or similar plant database API.

Critical features describe crucial functionality needed for minimum viable product. These features are higher priority than secondary features and will therefore be implemented first. Some secondary features may not be completed during the set development time (course duration).

3 Analysis

Caring for houseplants can be a demanding task, especially as the number of plants in one's home increases. Houseplants may originate from vastly different ecosystems and therefore have different needs that must be met in order for the plant to thrive in an environment that is foreign to it. Houseplants need to be watered, fertilized and possibly repotted on occasion. The frequency of these activities can depend on the plant itself, but also on other factors such as the current season or the humidity of your room. Therefore, the difficulty of caring for houseplants increases exponentially as their number increases.

This project aims to create a web app that makes these tasks manageable. The most important feature of this webapp is the ability to set reminders for your plants, to help keep track of duties like watering and fertilizing. These reminders can be set to only be active on certain time periods, which means the user will not have to manually keep changing their reminders when the seasons change.

Each plant will have its own history. History helps keep track of anything that happens to a plant, such as when it was last watered or repotted. History entries can also be made for irregular events such as moving the plant to a new location. Access to such history about a plant could help identify potential problems that it may have. For example, if a plant's history shows that its leaves have begun withering after changing location, it could indicate a lack of sunlight.

Additionally, plants can be marked as having diseases or pests. The web app can be used to manually track the severity of these issues so that the user may take appropriate action to fix them and see whether their approach is working.

One of the more difficult development challenges with this project is implementing a plant database API. Even though this is not considered a critical feature (see section 2), it would greatly enhance the user experience and increase the value that this web app can provide for its users. Data from such an API could potentially be used to autofill fields when adding new plants, but also to give the user various scientific information about any of the plants they own. Initially, the author has considered using Trefle API for this purpose, but this is subject to change depending on the project's parameters.

4 Retrospective

As of writing this, the webapp is finished and deployed to Azure. All critical features detailed in the second chapter have been implemented, however, none of the secondary features made it into the final build due to time constraints.

The planned vision and the final result are near identical, except for a few minor changes. Originally, it was planned that collections could have a type, however this option was removed due to a lack of practical use. History entry type and reminder type were given one parent entity called event type. This was done to compare history entries and reminder types between each other.

The users have two roles they can belong to – client and admin. Client has access to all the normal functionality (like creating plants, tags etc.), but only admin can create, edit and delete entries in certain tables (sizecategory, tagcolor, eventtype, remindertype, historyentrytype, pesttype, pestseverity). The client can use these entries when creating their own things (i.e tag should be assigned some color), but can not alter them in any way. The admin is limited only by foreign key constraints. This is done so that admin does not accidentally delete user entries.

Appendix 1 – Flow screen

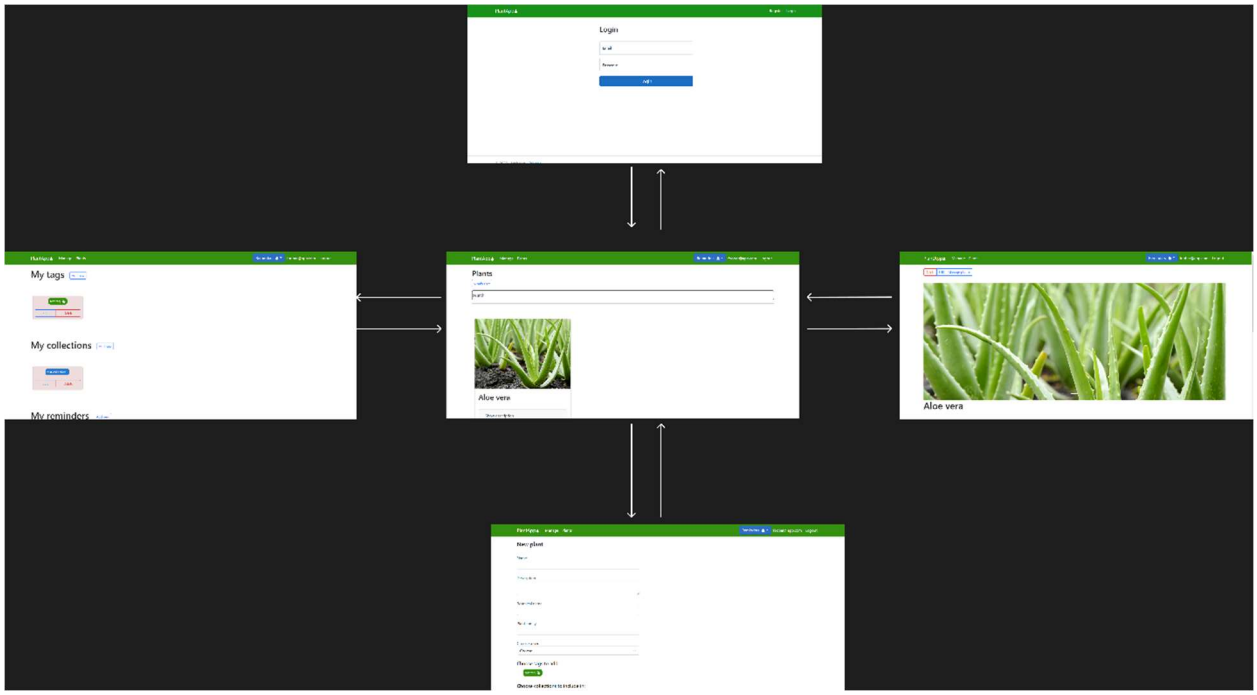

Diagram 1. Flow screens

https://www.figma.com/file/oqG0wZDzWLlutKkcpZbyGD/Houseplants?node-id=0%3A1&t=wyT98yWzR5Pn6cbC-1

Appendix 2 - Entity relationship diagram

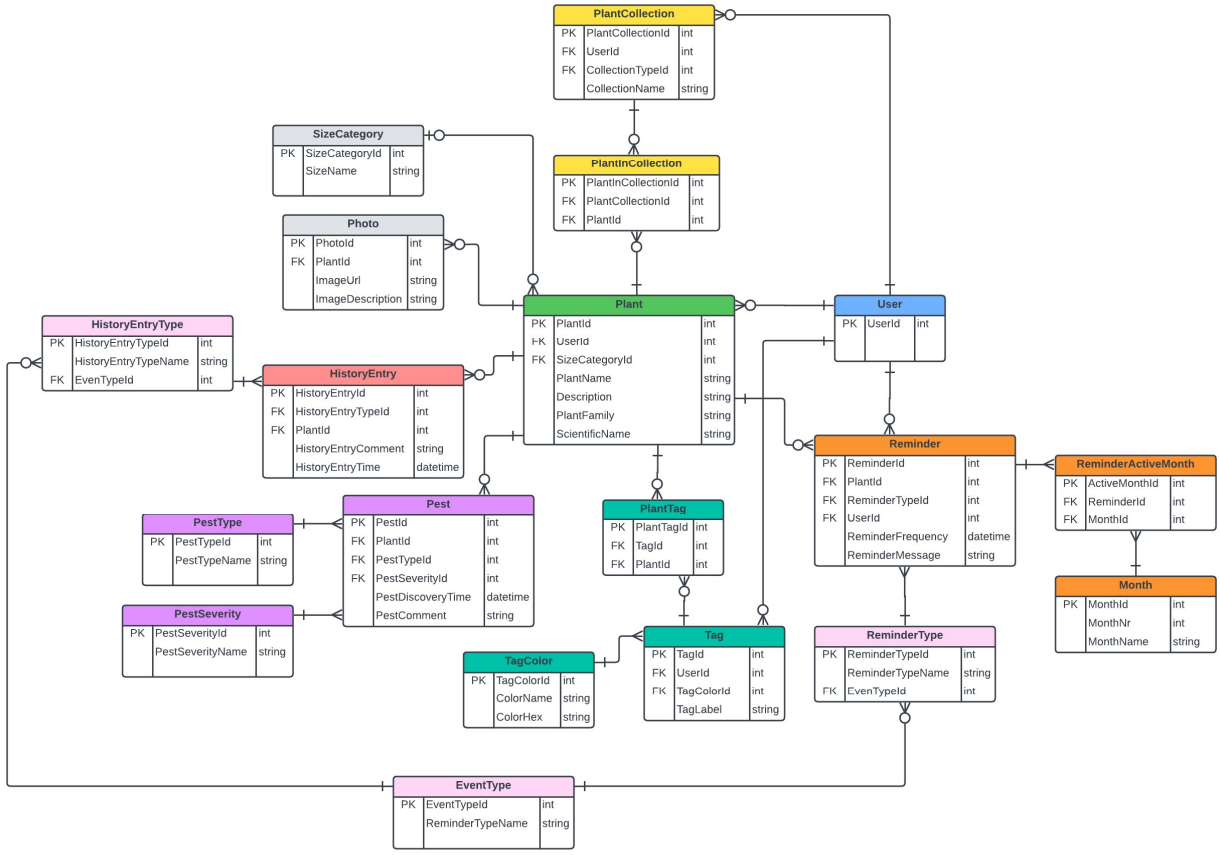

Diagram 2. Entity relationship diagram

https://lucid.app/lucidchart/65a4db27-fd9e-4e2e-b7d8-4158735658e1/edit?viewport_loc=-420%2C-211%2C3328%2C1578%2C0_0&invitationId=inv_8cc94817-9b1f-4a9d-b827-4a6c0d762c65