# Family Tree

# Goal: Create a family tree for a digital picture frame, a purposefully built processor specifically for the task.

# Tree Organization and Features

Family Members Organized with the oldest generations at the top, newer generations at the bottom. Tree can take different themes like members with a particular Surname or male – line descendants.

Types of tree organization:

Ancestry Chart

* Shows Ancestor of an individual, closely relates to a tree in shape.
* Wider up top than the bottom
* Information to include can be:
  1. Child Parent can include describing the lineage type i.e. biological, adopted, guardianship

Descendant Chart

* Descendants of an individual
* Narrowest at Top

Pedigree Chart

Resource: Guide for design <https://help.mycanvas.com/hc/en-us/articles/360000302586-Information-about-Ancestry-Family-History-books>

## Features

1. Display different tree organizations see section 1.
   1. Ancestry Chart
      1. Implement with names first
         * Future update implementing the rest of the Charts
      2. Future update start implementing with Pictures.
      3. Implementing step parents
2. A login feature
3. A Database for storing
4. Display a Timeline on the side of UI
5. Display a map with region family member lived

# UI/UX Design

## Navigation

Home Page

Opening:

1. Check to See if there’s a family tree
   1. If there is opens the last opened tree
   2. If not generic page that says start tree.

# Implementation

## Underlying container:

### Set

1. Binary tree with only two directions to directions, mother and father
   1. Can sort anyway by passing a comparator function.

### Classes

#### Person

#### Family Member

* Add image features to class

# Port to Embedded Device

# To do:

## Database

### Future Updates

1. Learn how to create a directory to store actual environment and database files
2. When generating charts (ancestry, descendant), figure out how to pull from information from cache without going back into DB.
3. Keeping database multi-threaded and atomic

## Fundamental Classes

### Name

1. Fix fullNameText()
   1. Logic of clearing fullname variable every time is called is not smart.
      1. Name will not change often
      2. Implement a way to detect if name changes and handles this fullnamevariable

### Relationship

1. Changed Relationship variables to strings
   1. Used to find actual username in database
   2. Change all other classes to reflect changes.