I) Database

- Account Model
- User Model
- Profile Model
- Planted Tree Model
- Tree Model

> Well all users have a profile it does not need a class + model only for the profile, but, since the description shows it will make it, it opens for more options in the future

II) Admin Pages

- New user creation allowing associating with an Account
- Listing and creation of Accounts
- Activiation and deactivation of Accounts

} BASIC CRUD

- Creation and visualization of Trees
  - ☑ Must allow to see the data of where it was Planted

Planted TRees

III) Template Views

- Login for Users that have been created by the Admin
  → only his !!!
- Logged user can see it's planted trees
- Information of the planted tree
- Add a planted tree

↳ Show all the ==planted trees== in the ==accounts== the ==user== it's a part off?

IV) REST API (We can do more stuff if + time)

↳ List all the ==planted trees== of the current ==user==

V) Test CASES

Case configuration:
- 2 ==accounts==
- 3 ==user== distributed between ==accounts==
- Some ==trees== planted by the ==users== at random

↳ Template test where the ==user planted trees== is being correctly implemented

↳ Template test where when trying to access other logged ==user planted trees== returns a 403 HTTP error

↳ Template test where the ==planted trees== of the ==account== the ==user== is a part of is being rendered correctly

↳ Unit tests for the ==user== `plant_tree()` and `plant_trees()` are working correctly and are creating the planted tree object correctly

Printed tree object (concept)