ECSE 321: Intro to Software Engineering

Project Deliverable#1

Requirements Document and Domain Model

Group 15:

Lea Akkari – 260674169
Mahdis Asaadi – 260663335
Mohammed Edris Jebran – 260742799
Reem Madkour – 260726480
Abed Al Rahman Atassi - 260720213

McGill University, Department of Electrical and Computer Engineering

Table of Contents

Requirements

Functional Requirements

ID:	REQ-F1
Function:	Add tree data and display on map
Description:	The TreePLE system shall allow the user to report a newly planted tree in an urban environment
Input:	Latitude and Longitude Coordinates, Municipality, Kind of Land Use, dimensions, the species of tree, date of survey, name of reporting person.
Output:	Add data of tree and set tree status to 'planted' and assign unique ID
Traceability:	Report Planting

ID:	REQ-F2
Function:	Mark tree as cut
Description:	The TreePLE system shall allow the user to report a newly cut tree in an urban environment
Input:	Latitude and Longitude Coordinate, name of reporting person, date of survey
Output:	Remove tree from map and set tree status to 'cut down', update name of reporting person and date of survey
Traceability:	Report Cutting

ID:	REQ-F3
Function:	Mark tree as diseased and update on map
Description:	The TreePLE system shall allow the expert user to report a diseased tree in an urban environment
Input:	Latitude and Longitude Coordinate, name of reporting person, date of survey
Output:	Mark diseased tree on map and change status to 'diseased', update name of reporting person and date of survey
Traceability:	Report Diseased

ID:	REQ-F4
Function:	Mark tree as to be cut down and update on map
Description:	The TreePLE system shall allow the expert user to report a tree as 'to be cut down' in an urban environment
Input:	Latitude and Longitude Coordinate, name of reporting person and date of survey
Output:	Mark tree to be cut down on map and change status to 'to be cut down', update name of reporting person and date of survey
Traceability:	Mark to be cut down

ID:	REQ-F5
Function:	Find tree on map given latitude and longitude coordinates
Description:	The TreePLE system shall allow the user to insert the location of a tree and find it on the map
Input:	Latitude and Longitude Coordinates
Output:	Take user to tree location on map
Traceability:	Locate Specific Tree

ID:	REQ-F6
Function:	Show trees of a given species on map
Description:	The TreePLE system shall allow the user to specify a species and find all corresponding trees
Input:	Species of tree
Output:	Only show specified trees on map
Traceability:	Locate Specific Species

ID:	REQ-F8
Function:	Show trees on map given status
Description:	The TreePLE system shall allow the insert a tree status and find all corresponding trees on the map
Input:	Tree status
Output:	Only show specified trees on map
Traceability:	Locate Trees in State

ID:	REQ-F9
Function:	Calculate Biodiversity Index
Description:	The system shall automatically calculate sustainability attributes such as biodiversity index which reflects the number of species in an area at any time.
Input:	All tree data
Output:	BioDiversity Index
Traceability:	Calclulate Biodiversity Index

ID:	REQ-F10
Function:	Calculate Canopy
Description:	The system shall automatically calculate sustainability attributes such as biodiversity index which reflects the number of species in an area at any time.
Input:	All tree data
Output:	Canopy
Traceability:	Calculate Canopy

ID:	REQ-F11
Function:	Calculate Carbon Sequestration
Description:	The system shall automatically calculate sustainability attributes such as carbon sequestration index which reflects the number of species in an area at any time.
Input:	All tree data
Output:	Carbon Sequestration
Traceability:	Calculate Carbon Sequestration

ID:	REQ-F12
Function:	List Trees in Municipality
Description:	The TreePLE system shall list all trees in the specified municipality
Input:	Municipality Name
Output:	All trees in specified municipality
Traceability:	List Trees in Municipality

ID:	REQ-F13
Function:	List Trees of a Certain Species
Description:	The TreePLE system shall list all trees of a specified species
Input:	Species Name
Output:	All trees in specified municipality
Traceability:	List Trees of Specific Specifies

ID:	REQ-F14
Function:	Update Tree Data
Description:	The TreePLE system shall allow the user to update tree data
Input:	Tree Data
Output:	Updated Tree Data
Traceability:	Update Tree Data

ID:	REQ-F15
Function:	View Map
Description:	The TreePLE system user to view trees on map
Input:	Tree Data
Output:	View current trees with current states
Traceability:	ViewMap

ID:	REQ-F16
Function:	Forecast
Description:	The TreePLE system shall allow the expert users to add what if scenarios and the system will generate predicted impact
Input:	Tree Data and Scenario Field Requirements
Output:	Predictions
Traceability:	Provide Forecast

Non Functional Requirements

ID:	REQ-NF1
Description:	The system shall be designed to differentiate the type of different users at all time as local residents can only cut down or plant trees within their own properties.

ID:	REQ-NF2
Description:	The system shall be designed to compile a report about sustainability attributes in what-if scenarios to help implement a plan for urban forest management by scientists and urban foresters at all time and the underlying provisional changes need to be persisted so that a report can be generated at any time.

ID:	REQ-NF3
Description:	The system shall be able to retrieve and store the longitude and latitude of a specific tree in less than 5 seconds.

ID:	REQ-NF4
Description:	The system shall be able to provide a systematic overview of all tree by listing all trees in a municipality, enumerating all trees of a specific species, plotting each tree on a map, and specifying the status of each tree to all users at all time.

Use Cases

Use case: Login
Goal: The local residents or scientist can log into the system which then loads the data from the database
Flow: Log in
Actors: Local Resident, Scientist.
Requirements: -

Use case: Update Tree Data
Goal: update different traits of a tree
Flow: Log in -> Update Tree Data
Actors: Scientist.
Requirements: REQ-F14

Use case: View Map
Goal: View current tree map
Flow: Log in -> View Map
Actors: Local Resident, Scientist.
Requirements: REQ-F15

Use case: Locate Specific Tree
Goal: Locate a tree chosen by resident/scientist
Flow: Log in ->View Map ->Locate Specific Tree
Actors: Scientist.
Requirements: REQ-F5

Use case: Locate Specific Species
Goal: Locate and view all trees from a certain species on the map
Flow: Log in ->View Map -> Locate Specific Species
Actors: Scientist.
Requirements: REQ-F6

Use case: Locate Trees in State
Goal: Locate all trees in a state on the map
Flow: Log in ->View Map -> Locate Trees in state
Actors: Scientist.
Requirements: REQ-F7

Use case: Report Planting
Goal: Resident or Scientist can report that they planted a new tree
Flow: Log in ->Report Planting
Actors: Local Resident, Scientist.
Requirements: REQ-F1

Use case: Report Cutting
Goal: Resident or Scientist can report that they cut down a new tree
Flow: Log in ->Report cutting
Actors: Local Resident, Scientist.
Requirements: REQ-F2

Use case: List trees in municipality
Goal: Resident or Scientist view trees in a certain municipality
Flow: Log in ->List Trees in Municipality
Actors: Scientist.
Requirements: REQ-F12

Use case: List Trees of specific Species
Goal: Resident or Scientist can view a list of all trees of a specific species they choose
Flow: Log in -> List Tree of Species
Actors: Scientist.
Requirements: REQ-F13

Use case: Calculate Attribute
Goal: Resident or Scientist receive a value for an attribute out of the available attributes.
Flow: Log in ->Calculate attribute.
Actors: Scientist.
Requirements:

Use case: Calculate Biodiversity Index
Goal: Resident or Scientist receive calculated Biodiversity Index.
Flow: Log in ->Calculate attribute -> Calculate Biodiversity Index.
Actors: Scientist.
Requirements: REQ-F9

Use case: Calculate Canopy
Goal: Resident or Scientist receive calculated Canopy.
Flow: Log in ->Calculate attribute -> Calculate Canopy.
Actors: Scientist.
Requirements: REQ-10

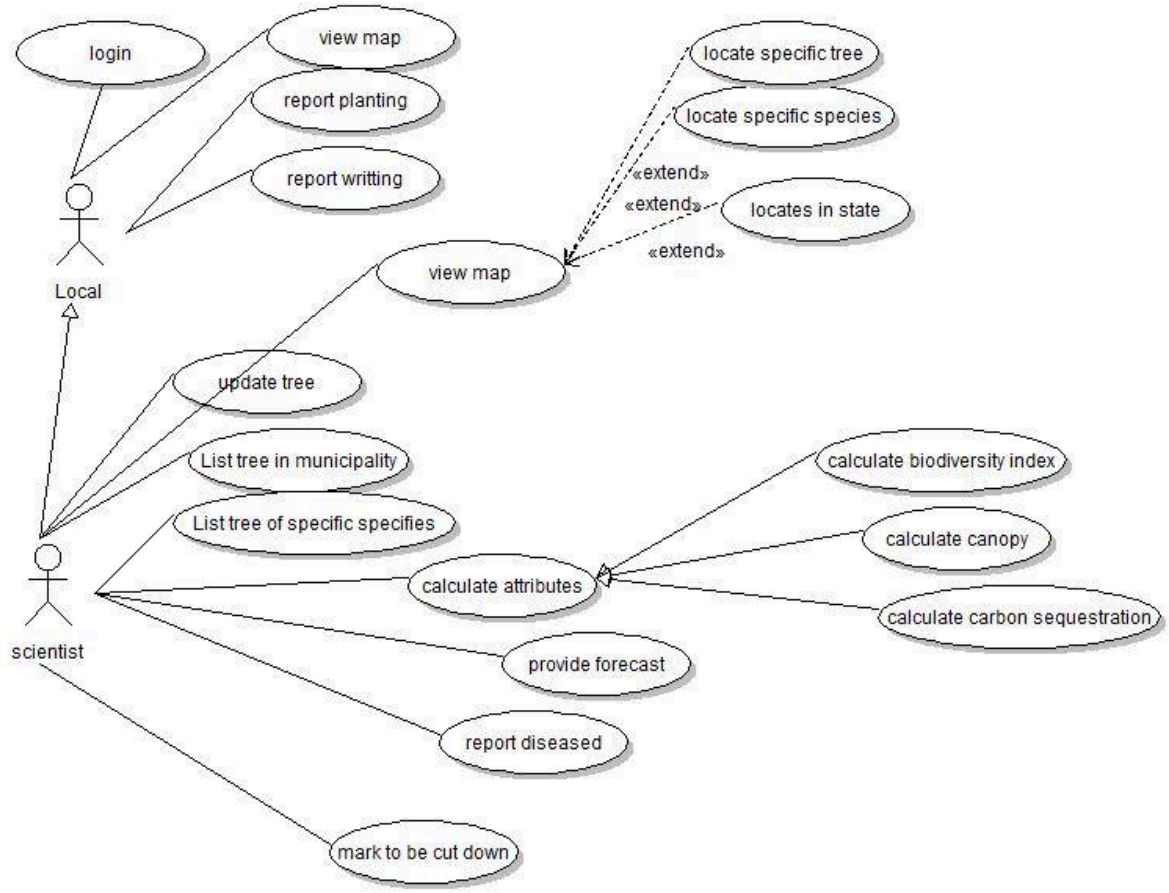
Use case: Calculate Carbon Sequestration
Goal: Resident or Scientist receive calculated Carbon Sequestration.
Flow: Log in ->Calculate attribute -> Calculate Carbon Sequestration.
Actors: Scientist.
Requirements: REQ-11

Use case: Provide Forecast
Goal: Scientist can provide what-if scenarios and the system will generate predicted impact.
Flow: Log in ->Report Planting
Actors: Scientist.
Requirements: REQ-F16

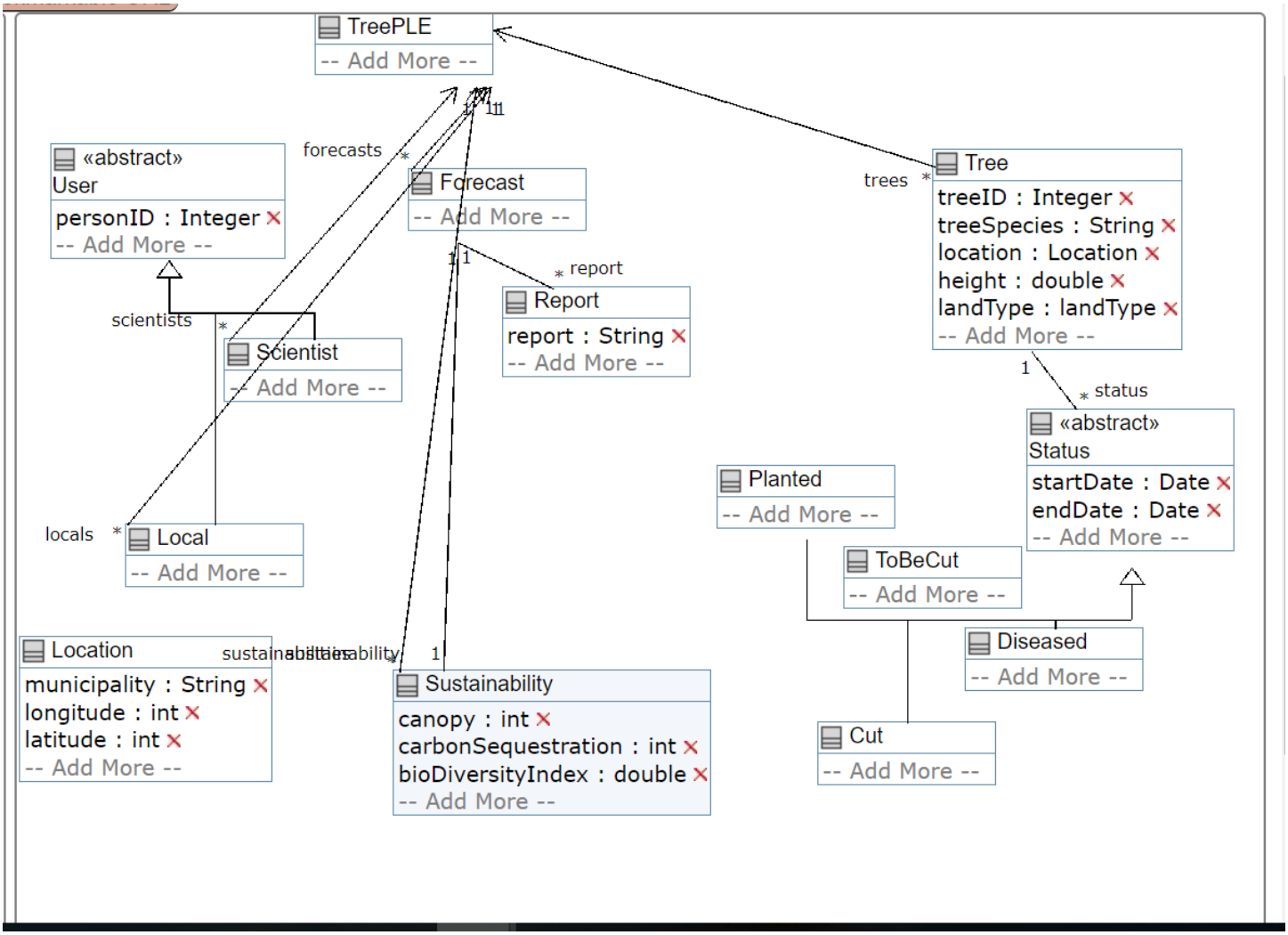
Use case: Report diseased
Goal: Scientists only can report trees as diseased
Flow: Log in -> Report Diseased
Actors: Scientist.
Requirements: REQ-F3

Use case: Mark to be Cut Down
Goal: Scientists only can mark trees to be cut down
Flow: Log in -> Mark to be cut down
Actors: Scientist.
Requirements: REQ-F4

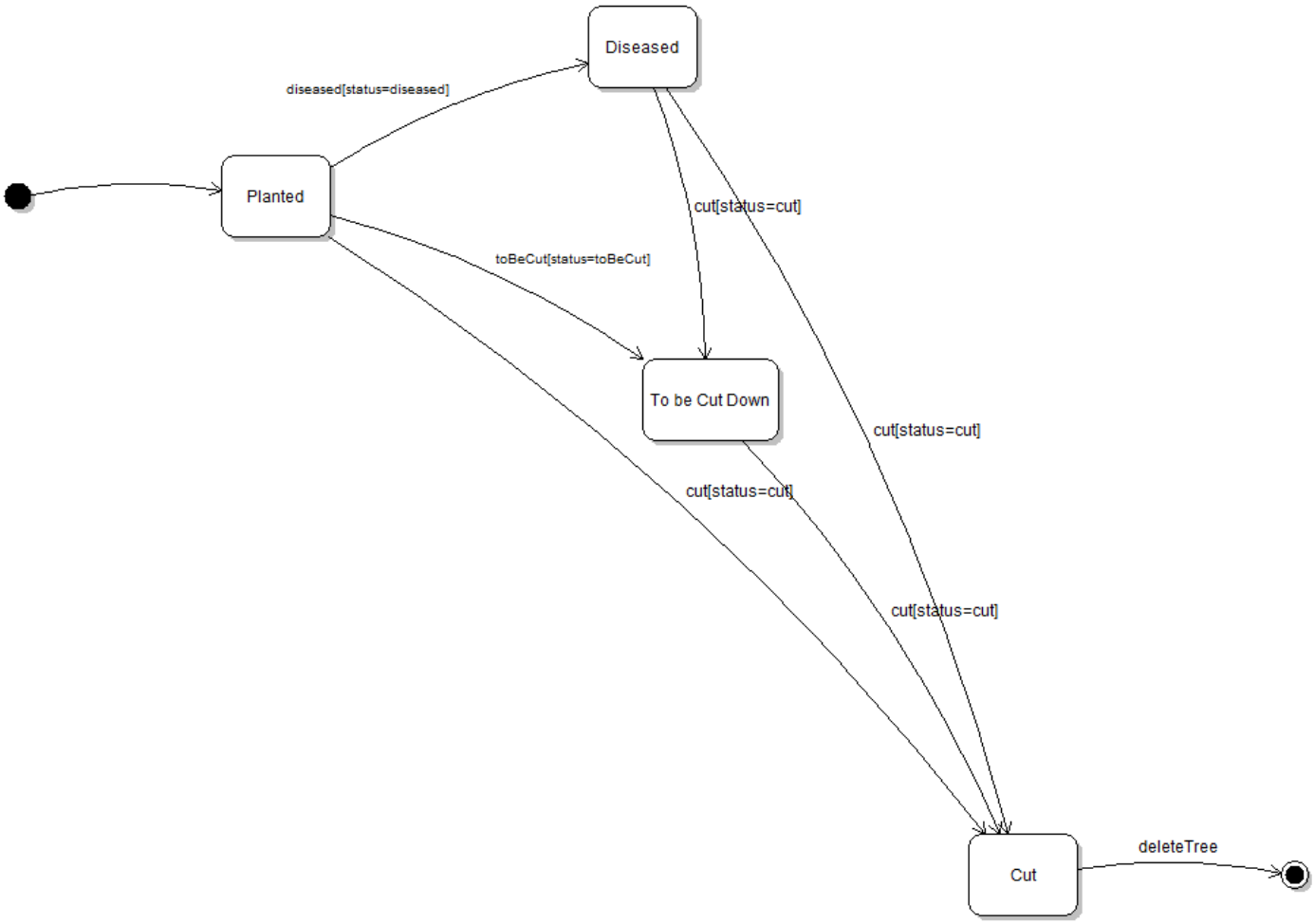
Use Case Diagram



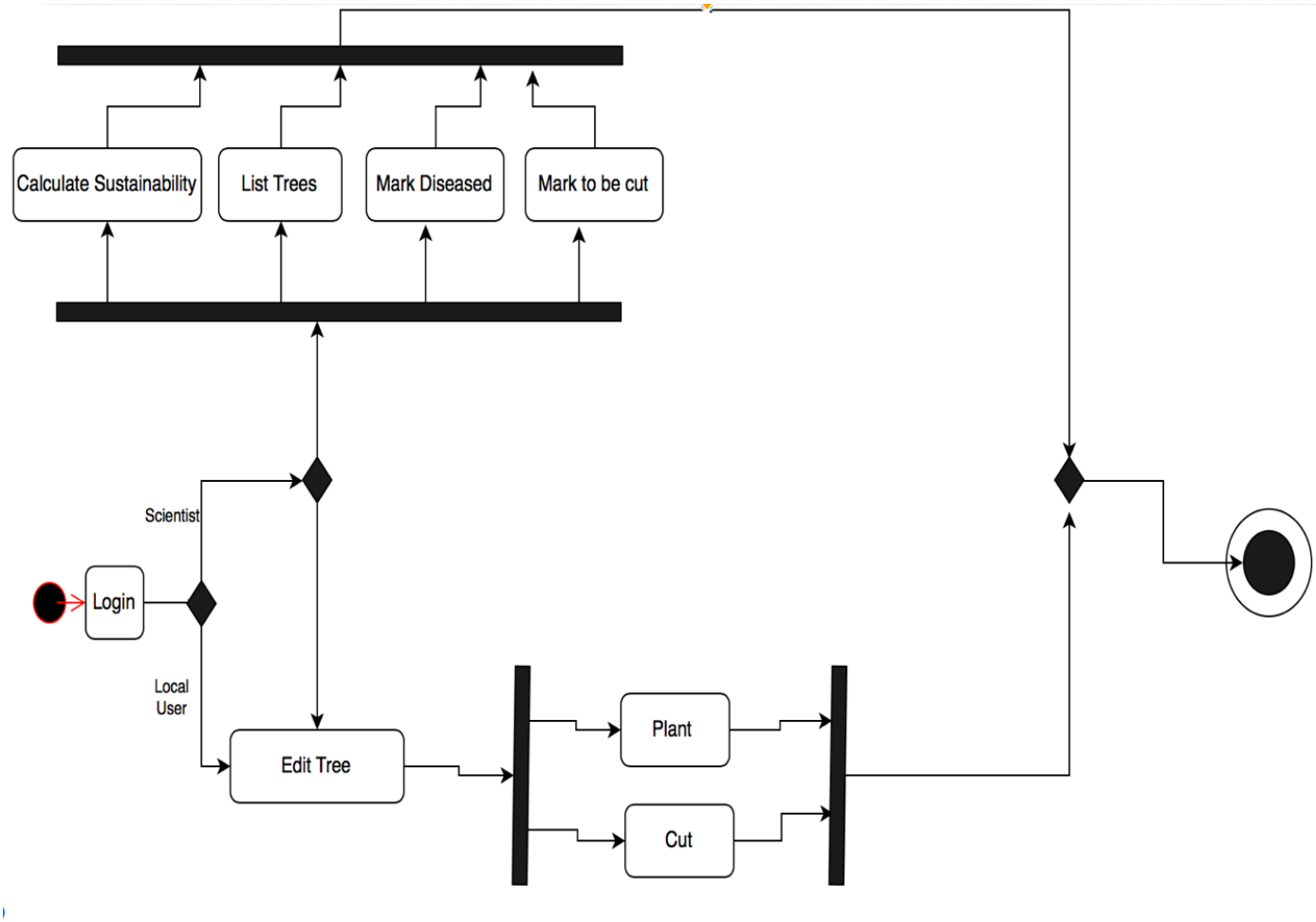
Domain Model



State Diagram:



Requirements-level activity diagram



Work Plan and Progress Report

Iteration 1: Requirements Analysis and Domain Modeling

Work Layout:

-Main Tasks:

Functional Requirements: Mahdis and Abed
Non-Functional Requirements: Lea and Edris
Use Cases: Reem
Activity Diagram: Lea and Mahdis
Domain Model : Edris and Abed

Work hours:

-Monday: 2 hours individually
-Thursday: 2 hours individually
-Friday: Group meeting 6:00-10:00pm (Functional requirements and Non functional requirements were done. Everyone was there and decided we should meet Saturday to finalize work, work on use cases, activity diagrams and domain model were started by the assigned individual)
-Saturday : group meeting 4:00 PM to 12:00 AM (The use cases and the diagram, activity diagram and the domain model were completed. The work for Deliverable 1 was finalized.)
Everyone participated and did their part, it was a very good and smooth meeting.
Iteration 2-6: Left to be done

Iteration 2: Design Specification and Prototype

Estimated time needed to complete 15hours, **should be done by February 22nd.**
Workload will be divided proportionally, 4hours of individual work and 11hours of group work.

- **Description of architecture, Block diagrams:**
- **Description of detailed design, class diagram**
- **Prototype implementations**
- **Implementation-level sequence diagrams**

Iteration 3: Quality Assurance Plan:

Estimated time needed to complete 15hours, **should be done by March 10th**.
Workload will be divided amongst the team members, 3 hours of individual work and at least 12 hours of group work, the work will be done spread over the week of February 27, so as to have more time if needed.

Iteration 4: Release Pipeline Plan

Estimated time needed to complete 13hours, **should be done by March 20th.**
Over the 10 days from March 1st to 20th ,the team will meet regularly to work together, if more work needs to be done, extra hours will be allocated to work.

Iteration 5: Presentation

Estimated time needed to complete 15hours, **should be done by April 1st.**
The team will allocate at least 15 hours to prepare the presentation, and get work done. 3-4 times a week the team will meet to work, prepare the presentation.

Iteration 6: Final Application:

Estimated time needed to complete 12hours, **should be done by April 8th.**
The team will meet regularly to finalize the project, make updates, test, once everything works, source code will be uploaded to GitHub repo.

UMPLE CODE

```
namespace ca.mcgill.ecse321.treePLE.model;

class TreePLE{
  1<-*Tree trees;
  1<-*Forecast forecasts;
   1<- *Local locals;
  1<-*Scientist scientists;
  1<-* Sustainability sustainabilities;
}

class Tree{
  autounique treeID;
  String treeSpecies;
  Location location;
  double height;
  enum landType{residential,institutional, park, municipal};
  lazy landType landType;
  1--* Status status;
}

class Location{
  String municipality;
  int longitude;
  int latitude;
}

class Status{
  Date startDate;
  Date endDate;
  abstract;
}

class User {
  abstract;
  autounique personID;
}

class Scientist{
  isA User;
}

class Local{
  isA User;
}

class Planted{
  isA Status;
}

class Cut{
  isA Status;
}

class ToBeCut{
```

```
  isA Status;
}

class Diseased{
  isA Status;
}

class Forecast{
1--1  Sustainability sustainability;
1--* Report report;
}

class Report{
  String report;
}

class Sustainability{
  int canopy;
  int carbonSequestration;
  double bioDiversityIndex;
}
```