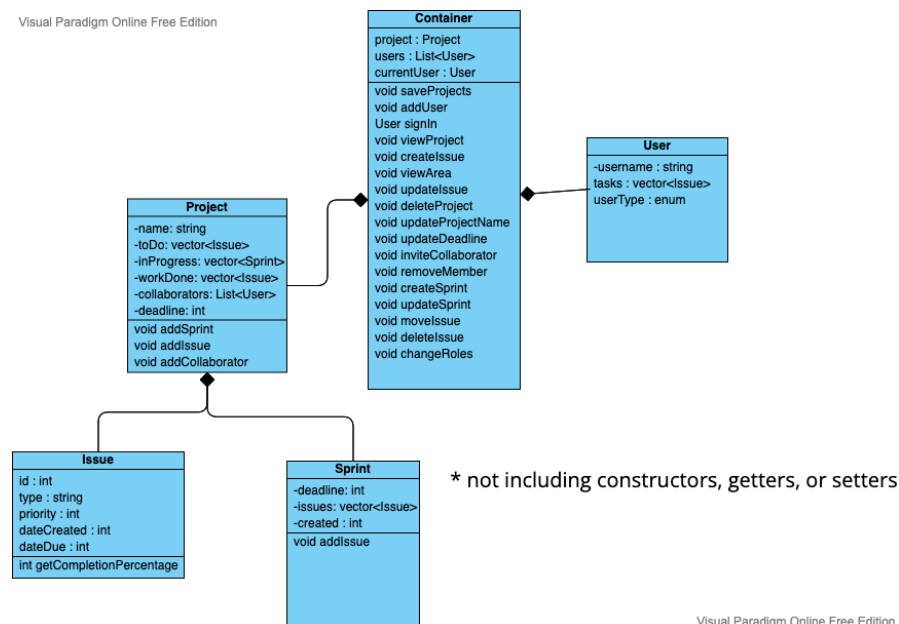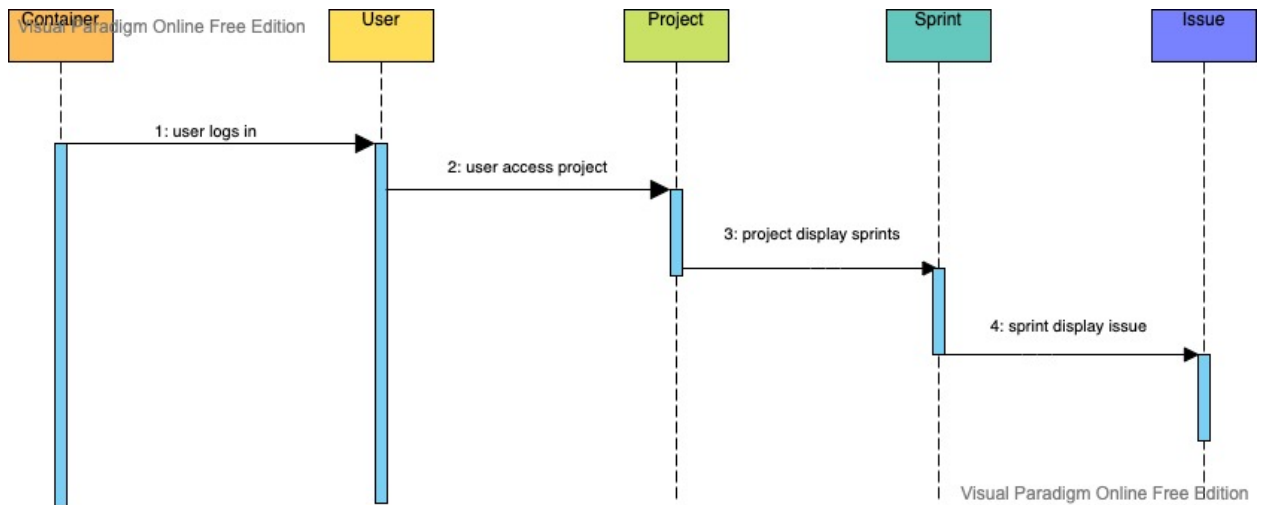# Final Report

Testing :

For testing purposes, we first tested each operation and each component of the system as a
project member, lead, and owner. To make the testing more efficient, we created an overloaded
constructor of the container class which would take in a Project pointer, a vector of User
pointers, and a User pointer to represent the current User. This allows us to recreate a stage of
a project when a user is interacting with our issue tracking app. When we passed in a project
member as the current user, we tested to make sure the user didn't have the same
responsibilities as a lead or owner (ex: deleting a project). In the code itself, we checked to
make sure the current user was logged in and wasn't a member whenever a responsibility of a
lead or owner was called. For testing when the current user was a lead, we made sure they
could create and update sprints, invite other users, and change properties of the project. For
testing when the current user was the project owner, we made sure they had all the same
responsibilities of a member and lead with the extra ability to delete the project. After creating
the main menu options, we also tested that the menu options only work when the correct input
is provided by the user by running the program with invalid inputs as well.

Diagrams :

Container

project : Project
users : List<User>
currentUser : User

void saveProjects
void addUser
User signIn
void viewProject
void createIssue
void viewArea
void updateIssue
void deleteProject
void updateProjectName
void updateDeadline
void inviteCollaborator
void removeMember
void createSprint
void updateSprint
void moveIssue
void deleteIssue
void changeRoles

User

-username : string
tasks : vector<Issue>
userType : enum

Project

-name: string
-toDo: vector<Issue>
-inProgress: vector<Sprint>
-workDone: vector<Issue>
-collaborators: List<User>
-deadline: int

void addSprint
void addIssue
void addCollaborator

Issue

id : int
type : string
priority : int
dateCreated : int
dateDue : int

int getCompletionPercentage

Sprint

-deadline: int
-issues: vector<Issue>
-created : int

void addIssue

* not including constructors, getters, or setters

Container	User	Project	Sprint	Issue

1: user logs in

2: user access project

3: project display sprints

4: sprint display issue

CRC Card

User	
Get Name	Project User

Container	
Save Project Add Users Get Project Get User Create Issue View Project View Area Update Issue Update Name Update DL Invite User Remove User Create Sprint Update Sprint Move Issue Delete Issue Delete Project	Project User

Project	
Add Issue Add User Add Sprint	User Issue Sprint

Issue	
Get Description	

Sprint	
Add Issue	Issue