**Student Project Sharing App**

**Table of Contents**

**Student Project Sharing** [**app**](#_w19eu4inkf3t) **2**

[Background](#_t1hhpg4pd4ic) 2

[Glossary](#_7i88o7dcno5u) 3

[Goals](#_e7dv482vk68c) 4

[Speculation about Desirable Solution](#_mfeqkhf30cnh) 4

[Non-Goal](#_kw0078tw2u9j) 5

[Actors](#_dcy2c9w794ej) 5

[Models](#_jwrxxfdbtbhx) 5

[Software Architecture](#_7g9pdg2ggvob) 6

[Use Cases](#_3a98l0b27xf8) 6

User 6

Organization

Community Admin 6

Progress 7

Milestone 7

## **Goals**

The goals of the app are to increase the visibility of student initiatives within the community in terms of software projects. The application will provide current UofT students a platform to discover software initiatives happening in the community, and apply to be apart of that initiative. The initiatives will appear in the form of software projects provided by students, non-profit organizations, clubs, or UofT staff within the community. The platform will provide students with the best access to discover software projects that will enhance their learning and collaborations skills. The system should be usable by all members within the community at UofT.

#### **Non-Goal**

It is not a goal of this project to provide students with projects, or help them in any way to build their existing project. Community efforts would be required to support students in learning new skills and build their projects.

## **Introduction**

Finding individuals with a similar passions and interests is an ambitious endeavour when there are over 1000+ CS POSt students at UofT.

The proposed system would allow members of the software community at UofT to search and filter for projects worked on by their peers, post interested projects onto the platform for others to join, and discover other users who have similar interests or skill sets.

The system should authenticate users, such that users are confident they are not dealing with imposters. It is important for users to know what existing projects exist within the community that others are working on. The system should be able to send a configurable level of reminders (such as emails) to notify users on incoming applications or different application processes. Project owners should easily create, review and approve user applications to work on their project. Ideally, auto matching students to projects would make it easier for students to find projects their skills and interests tailor to.

The system will adhere to the [Bezos Dictum](https://docs.google.com/document/d/e/2PACX-1vQdzMAbRnBDEav3q6GMizxIcNfuTHEC1oHXKnhTyUgv9kDcyCZuNvobSSVKN7n6d9jDlFe6BYYtyigv/pub), so essentially all the functionality will be exposed as a set of endpoints consumed by some combination of user interfaces (eg web and/or mobile). The architecture of your system must reflect security issues.

## **Background**

Currently, UofT students are resorting to individual projects, finding projects from personal connections or online resources. However, since students have a variety of technical experience, it is often difficult to find peers who have similar skills to work with or find a willing mentor to learn from. Project-collab provides these students with the platform to connect with each other, along with resources to get students started working with new technology or projects they are interested in.

On a recent community survey conducted, we asked the community of computer science students, how the CSSU/CS department/CS community could support them in aspects of career and side projects. The results were quite reassuring: 12/49 responses wanted support for project related work, through a platform to share and discover personal projects. The community seems to have validated the project for us, bringing up the need for the platform we have envisioned.

The raw data is viewable in an excel file with the following link:<https://docs.google.com/spreadsheets/d/1ka9U_uB1w1ItPvCYsGl3NL-Egw5o3WS78GTWiSnlbFQ/edit?usp=sharing>

### **Glossary**

A *user* is any account created in the app.

A *project* is any posting that a *user* creates that describes the project they are a part of.

A *project owner* is the user that created the project.

A *tag* is a category in which a *project* belongs in.

A *community admin* is a type of *user* who can remove a *project*.

*Endpoint* is a route, or set of routes, that expose a service online. Typically HTTP (probably HTTPS) is used to describe the format of the messages. Typically, arguments are passed to endpoints as JSON. One can think of each endpoint as a part of the API offered by software as a service.

##### **Speculation about Desirable Solution**

A typical approach would build a software service (endpoint), probably behind a proxy server, that would serve student-facing apps routes for authenticating, project creation and tag filtering, etc.

In a perfect world the implementation would be containerized or otherwise easy to deploy.

## **Actors**

1. Project Posters (Examples):

* Experienced student planning to build large projects and are looking for students.
* Inexperience student looking for explore new areas of expertise with others.
* Professors looking for students to help with research projects.
* University, non-profits organizations looking for students to work on social projects.

1. Project Searchers (Examples):

* Inexperienced students at UofT students who are interested in collaborating on software projects, gaining hands-on experience outside of class, and looking for a platform to learn, discover and demonstrate these skills to employers. A typical users from this group could be a 1st year Computer Science student, looking to expand their skills to land an internship after the summer.
* Experienced students looking to learn new technologies through projects that focuses on their technology of interest.

## **Models**

* *Users (Student, Organizations, Community admins)*
* *Projects*
* *Tags*

### **Software Architecture**

As per Bezos dictum posted. Backend APIs must be exposed as endpoints returning JSON. Distinct endpoints for Project management(Create, Update, Delete, Search, Filter), User management(Creation, Delete, Authentication, Update, Search) are required.

Front-end technologies:

* Webpack, Angular 7

Back-end technologies:

* Node, Express, Sequelize, Postgres

## **Use Cases**

### **\*** Bolded use cases are features suitable for CSC301, the rest are for CSC302

### **Site Visitors**

1. Browse Projects without logging in
2. Account creation for Student
3. **Account creation for Organization**
   1. **Login process must provide more information and must be verified by community admin**
4. **Any attempts to user features aside from browsing projects will be prompted to login** 
   1. **Such as leaving comment and following user**

### **Users/Organizations with Account**

1. Everything a site visitor can do plus
2. Create/Update profile
3. Delete Account
4. Create/Update/Delete projects
5. **View project details, including rough detail of project owner, comment.**
6. **Leave a comment on project a project that’s not their own**
7. **Users that own a project should be able to delete comments on their project**
8. **Recommend users for other projects**
9. **Follow other users**

### **Community Admin**

1. **Cannot be created via login, account must only be created in the backend**
2. **Everything an User/Organization can do plus**
3. **Remove projects from a community that are inappropriate**

## **Progress**

Currently, the MVP consists only backend APIs that allow for

* User account registration
* User Login Authentication
* Project creation (any type of user)
* Project approval/Rejection (community admin only)
* Application to projects (any type of user)
* Application to projects Approval/Rejection (admin admin only)

## **Milestones & Timeline (Road Map)**

**In CSC302 (Developing core infrastructure)**

The following features are the core of this project proposal for CSC302:

* Front-end functionalities for:
  + User creation, login
  + Routing
  + Project creation, update, deletion process
  + User profile creation, update, deletion process
* Tag system to enhance project search, filter and enable recommendation
  + Filter Projects via tag
  + Recommend projects to users based on user skills
* Eliminate Entry Barrier
  + Allow any site visitors to browse and search for projects

**Timeline** (During 302)

* February (Vertical slice)
  + Complete the following use-cases:
    - Any site visitor can browse available projects
    - Any site visitor can create a student account
    - Students can login
    - Students can create/delete projects
    - Students can add/update tags to projects
  + Front-End:
    - Landing page, Display all projects
    - Login/Registration for student type user
    - Create project pages
    - Add search tags
      * Start with fixed number of tags, meaning users cannot create and add their own tags
  + Back-end:
    - Create API endpoints for tag creation, update, search
    - Update database schema to incorporate tags and association
  + Testing:
    - Basic UI tests for Front-end pages
      * e.g Check if main components are loading
    - API tests
      * Ensure api requests return expected results
      * Basic condition testing
  + Others:
    - 2 2-week sprint cycles
    - Setup Continuous deployment process
    - Documentation Write up
      * Project structure
      * Database design/tables
      * API endpoints update
* March
  + Complete the following use-cases:
    - Users are able to create/update personal profiles
      * With description, skills (tags) and interests (tags)
    - Users logged in with at least 1 tag in either their skill or interests will have their discover page projects ranked in the following way:
      * Projects that best match their skill and interest will appear on the top of search results with a % of match
    - User profiles are public to all users that are logged in
  + Front-end:
    - Profile pages
      * Update
    - Tag display for user
      * add/delete tags when updating profile
  + Back-end:
    - Recommendation system
      * (Optional) built as a microservice
      * Basic AI algorithm to provide recommendations
    - API endpoints for suggestion algorithm
    - API endpoints update for user-tag association
    - Database schema update for users
  + Other:
    - Update documentation
    - If user testing is conducted, additional features may be included
    - 2 2-week sprint cycles

## **For CSC301**

The 301 team will be asked to focus on the Networking features and peripheral features.

* Features that aim at enhancing the networking aspect
  + Allowing users to follow projects
  + Have a way to display relationship between users using the following information
    - View to see what project a user has followed
    - View for project owner to see who has followed/favorite their project, and a numbered
  + Allow users to leave comment under projects
  + Allow everyone else to view comments when viewing the project
  + Allow users to follow/favorite projects
  + Notification system (for follows/favourites)
    - Have somewhere to view notifications
* Others:
  + Login and Registration for Organization user-type
  + Creation process of Admin type users
  + Validate Users
  + User that are logged in can create comments under project details
  + Admins deleting projects
  + Inappropriate comment flagging system
* Make sure tests are included for any new API endpoints.
  + Conditional Testing
* UI tests should include:
  + Creation of page components
  + JS functions
* Note:
  + Team will not be asked for the following functionalities
    - Create project
    - User Profile
    - Project Filter, Search & Deletion from project owner