1. Introduction

Concept:
Imagine a bright new freshman to SFA. Full of energy but has nowhere to put it. Now obviously one of the first few ideas is to join a club. But then the question must be asked, where to search for these clubs? That's where this application comes in. It allows students to view the full list of clubs, but it also allows them to search for and be informed on which clubs might be best for them. A good portion of students may already have a club in their head, so they'll go search them in the application and then the story ends there, but we also can take of people who may not know what exactly what they want. This application allows for a student to take a quick questionnaire that ask them about their preferences and based on these preferences; we present a list of clubs they may enjoy. It's a simple yet effective way of getting students more informed about groups on campus.

Scope:
The application will:
Retrieve a list of organizations (clubs) from the presence API and display each organization's name, description, categories, meeting time, meeting location, member count and cover image. The API returns those fields as shown in sample data where each object includes the organization name, description, meeting time, meeting location, photo URL and categories.

Allow users to browse the list of clubs and view details for a selected organization, including contact information and a link to join the club through the presence site.

Provide search and filter functions so the user can quickly locate clubs by name or by categories/tags.

Present a questionnaire to gather students' interests and preferences and compute personalized club recommendations based on category matches and keywords.

Operate entirely in the client browser without requiring user account. Preferences are stored locally and resets when the session ends.

The application will not manage membership on the presence platform; when a user decides to join a club, the app will redirect them to the club's official presence page. Administrative functions (creating or editing the clubs) and authentication are out of the scope.

Functional Requirements (FR)
For clarity, each functional requirement is labeled FR.X. Functional requirements describes what the application must do.

FR.1. Organization list display

Description: The application shall retrieve a list of organizations from the presence API and display each entry's name and cover image on the main page.

Rationale: Students need way to quickly browse available clubs and see basic information

FR.2. Questionnaire and personal recommendations

Description: When a user selects an organization from the list, the system shall display detailed information, including:
Full description and mission statement (from the descriptum field).
Categories/tags

Regular meeting and time and location (if provided)
Member count and whether the organization is new (from member count and new organization).
Contact information and links contained in the description.
A Join button/link that opens the organization's page on the present site.

Rationale: Students need comprehensive information to decide whether a club fits their interests and to initiate membership.

FR.3. Search/Filter by name and tags (category filtering)

Description: The application will be able to filter clubs by search query to search for clubs by name or tags.

Rationale: Students need way to quickly filter available clubs and see basic information.

FR.4. Organization detail view

Description: When a user selects an organization from the list, the system shall display detailed information, including:
Full description and mission statement (from the descriptum field).
Categories/tags
Regular meeting and time and location (if provided)
Member count and whether the organization is new (from member count and new organization).
Contact information and links contained in the description.
A Join button/link that opens the organization's page on the present site.

Rationale: Students need comprehensive information to decide whether a club fits their interests and to initiate membership.

FR.5. Scrape info from API

Description: The application will be able to get information from SFA's API that has information on all the clubs registered.

Rationale: To do any task needed for this application, information must be obtained about the Clubs. So, using the API allows us to easily obtain the information and display the relevant information for the user, whatever this use for it may be.

FR Case Scenario Template

FR.1 View List of Clubs

Goal in Context: The User has the ability to see the list of all clubs.

MAIN SUCCESS SCENARIO

1. User clicks on the link to see list of clubs.
2. Page loads names of all clubs.
3. User views said list of names.

FR.2 Take Questionnaire

Goal in Context: The User has the ability to take a questionnaire to help find a possible club for them.

MAIN SUCCESS SCENARIO

1. User clicks on the link to the questionnaire.
2. A list of questions is displayed.
3. User answers all relevant questions.
4. User submits answers.
5. Based on answers, a list of clubs shows matching user's answers.

FR.3 Search For Club

Goal in Context: The User has the ability to use a search query to search for clubs by name or tags

MAIN SUCCESS SCENARIO

1. User clicks on link to search system.
2. User clicks on search query bar and enters characters.
    a. User may also search based on tags as well, and click on which tags for a club to have
3. User submits search query
4. Page displays results based on user's submission

FR.4 View Club Profile

Goal in Context: The User has the ability to view an individual club's profile

MAIN SUCCESS SCENARIO

1. User clicks on link to see list of clubs
2. User click on link to selected club they wish to view
3. Page loads information of club
4. User sees displayed information on said club

FR.5 Scrap Information from API

Goal in Context: When the User takes an action such as view clubs, the application will ping the API to get the information needed.

MAIN SUCCESS SCENARIO

1. User clicks on club list
2. Application pings API for relevant information
3. API returns relevant information