**Project Abstract**

This application, **StudyScapes**, will allow users to **login** to their account and view the Simon Fraser University campus as a virtual map. Login **views** will differ for faculty and students. This uses the **Web API for Google Maps** for location and time updates. As a **real-time feature**, this application will use **Socket.io** to update requests made to professors or students for meetings and study groups in real time. This application enables quick meetups and easy navigation for users. The application is not just practical, but fun! There are minigames incorporated to help the user kill time while waiting between meetups (they can even battle their professors!).

**Customer**

**Faculty:** Teachers and TAs at Simon Fraser University that would like to interact in a convenient way with students, plan meetings, coordinate meeting locations, schedule or find events on campus, and enjoy minigames.

**Students:** Students at Simon Fraser University that would like to communicate with all their professors or TAs in one convenient application. They would like to meet with other students or faculty at SFU, get assistance in finding their way around the school, find specific buildings or rooms, schedule or find events on campus, and meet other students or just have fun through minigames.

**Competitive Analysis**

What makes StudyScapes so unique is that it applies specifically to the Simon Fraser University Burnaby Campus. It incorporates ideas from many other successful programs into one main application for ease of access. Although Google Maps is useful, it does not provide a detailed view of the Burnaby Campus. StudyScapes will elaborate on the map with this campus in mind, allowing for much more extensive interaction and location finding around the University. SFU Snap was an inspiration for this, as it helped students to search for and travel between rooms and buildings across campus. StudyScapes expands this to allow for scheduling events and meetings around campus with students and faculty alike. It also provides ice-breakers for students in the form of minigames and different views for faculty and students, so that each user sees the information that directly applies to their own needs.

**Main Features - Epics**

**Map:** Provide an interactive map that updates to user location so they can find other users and locations on the Simon Fraser University Campus.

**Meetings:** Enable faculty and students to easily interact and meet up for studying or course/career discussion.

**Events:** Have events held by clubs, faculty or student societies show on the map as icons in real-time, so users can quickly learn about the activity on campus.

**Minigames:** Allow students and faculty to interact in mini-games with those nearby based on location. This would allow users to play small various games that can act as ice-breakers between their fellow peers.

**User Stories**

**Actors**: **Faculty** (Professors and Teaching Assistants) that can view and cancel meetings. They can also interact with students, view the map, view and schedule events, and play minigames. Students can view, request, reschedule, and cancel meetings with Faculty and other students. They can view and schedule events, view the map for specific rooms and buildings, and play minigames.

**Current Iteration Stories**

**Name/Description:** A new student is lost on the Burnaby campus

**Actors:** New student named ‘Sally’

**Triggers/Preconditions:** Sally logs in to StudyScapes and opens the map page to find her current location and destination

**Actions/Postconditions:** Display a campus map showing her current location and chosen destination (potentially with a path)

**Acceptance Tests:**

* Location services are enabled
* Current location can be found and display
* Destination is valid and can be displayed

**Iteration:** Started in *001*

*---*

**Name/Description:** A student wants to meet with another student

**Actors:** Multiple students named ‘Maddie’ and ‘Bud’

**Triggers/Preconditions:** Maddie accesses the meetup page and schedules a meeting with Bud

**Actions/Postconditions:** Displays a form for Maddie to specify the meeting location and time, then sends a request to Bud

**Acceptance Tests:**

* Meetup form is displayed properly
* Bud is an existing student and a request can be sent to him
* Meeting location is valid and can be chosen
* Form can be submitted
* Meeting data is stored in the meetup table

**Iteration:** Started in *001*

**---**

**Name/Description:** A professor wants to review their scheduled meetups

**Actors:** A faculty member, Professor Abe

**Triggers/Preconditions:** Professor Abe views the meetups page on their account

**Actions/Postconditions:** Displays a table that only lists the meetings including Professor Abe. Each row shows the corresponding student, the requested room, and the requested time.

**Acceptance Tests:**

* Meetup table is displayed with just Professor Abe’s meetings
* Valid rooms and times are displayed
* The corresponding student exists
* Finished/cancelled meetups are not displayed

**Iteration:** Started in *001*

**---**

**Name/Description:** A professor wants to cancel a scheduled meetup

**Actors:** A faculty member (Professor Bee) and a student (Erik)

**Triggers/Preconditions:** Professor Bee chooses to cancel her meeting with Erik from her meetups page, by clicking the cancel button next to the listed meeting

**Actions/Postconditions:** The meeting is listed as cancelled on the table (greyed out) and Erik is notified of this cancellation on his meetups page.

**Acceptance Tests:**

* The isCancelled attribute for that meeting is set to true
* The row is greyed out and the cancel button is no longer clickable
* Erik is able to view the meetup as cancelled through his meetups page
* The scheduled meeting remains in the database

**Iteration:** Started in *001*

**---**

**Name/Description:** A student responds to a cancelled meeting

**Actors:** A faculty member (Professor Bee) and a student (Erik)

**Triggers/Preconditions:** Erik wishes to view the meeting that was cancelled by Professor Bee, and confirm the cancellation

**Actions/Postconditions:** Erik clicks ‘confirm cancellation’ from the meetups page and the meeting is removed from both his and Professor Bee’s meetups

**Acceptance Tests:**

* The meetup (row) is removed from the *meetup* database
* Erik’s page is refreshed to no longer show the meeting
* Professor Bee can no longer see the meeting on her meetup page

**Iteration:** Started in *001*

**Future Iterations**

Name:

Actors:

Triggers/Preconditions:

Actions/Posconditions:

Acceptance Tests:

Iteration:

**User Interface Requirements**

![A close up of text on a white background

Description automatically generated]()

![A close up of text on a white background

Description automatically generated]()

![A close up of a sign

Description automatically generated]()

![A close up of a sign

Description automatically generated]()

![A close up of a sign

Description automatically generated]()

**Meeting 2 Overview**

* Discussion on separating room finder from Google Maps API
  + More research into SFU’s Room Finder and Esri
  + Use the Google Maps API for finding nearby students for minigames
* Decision to add an Admin user
  + Admin can view all database tables

**URLs**

**GitHub Repository:** *https://github.com/Guojiaxi/sfu-cmpt276proj.git*

**Heroku link:** *https://cmpt276proj-jlguo.herokuapp.com/*

**Heroku Git Link:** *https://git.heroku.com/cmpt276proj-jlguo.git*