SWEN90007

Part 1A Report

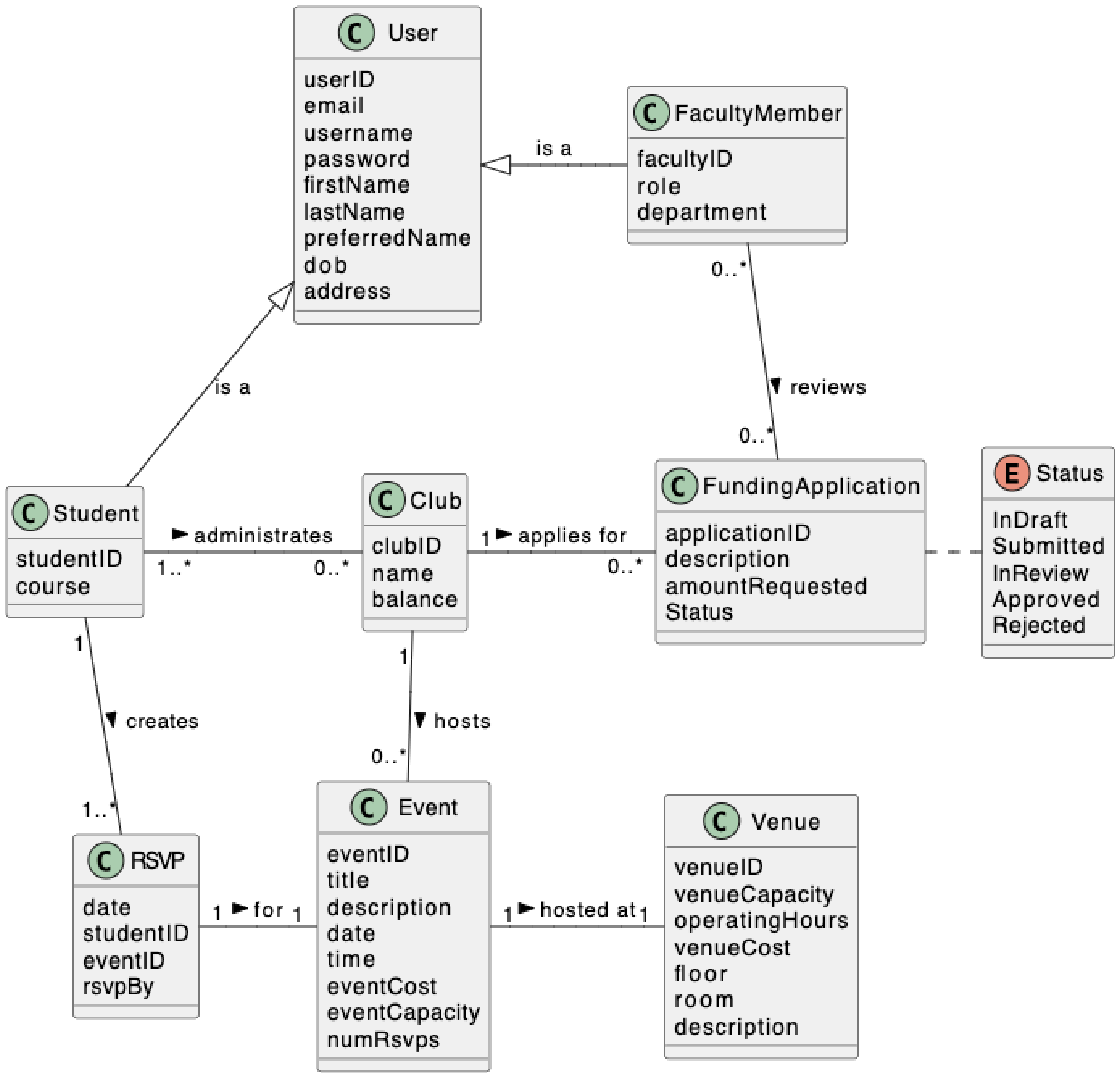
Team Details:

Team Name: BrogrammerBrigade

Name	Student ID	Unimelb Username	Github Username	Email
Kevin Wu	993272	wukw	kevwu99	wukw@student.unimelb.edu.au
Henry Hamer	1173633	hhamer	hentity	hhamer@student.unimelb.edu.au
James Launder	993934	launderj	JamesLaunder	launderj@student.unimelb.edu.au
Kah Meng Lee	940711	kahl3	KahMeng2	kahl3@student.unimelb.edu.au

Release Tag: SWEN90007_2024_Part1_BrogrammerBrigade

Domain Model



Domain Model Description

<u>User</u>

This is a parent entity representing a user of the system, either faculty or student. It includes some basic attributes such as an email, username, password and personal details.

<u>Student</u>

This entity represents a student using the system and inherits from User. As well as the attributes inherited from User, the ‘course’ attribute records which course the student is enrolled in. We decided to give students and faculty separate student and faculty IDs as well as their inherited user IDs, as this is common in similar systems.

Faculty Member

This entity represents a faculty member, who will use the system to review funding applications. As well as the inherited attributes from User, their role and department is also recorded.

RSVP

An RSVP instance represents an RSVP from a single student to a single event. The ‘rsvpBy’ field will be used to keep track of the original creator of the RSVP in the case where there is an RSVP for multiple students.

We included numRSVP as an attribute, as this allows us to get the number of attendees without querying the entire database.

Another option the team considered was to have an instance for each submitted RSVP, with extra attendees in a list attribute. However, we chose to have an instance for each individual student because we anticipate that this will speed up operations such as querying whether a student has already RSVP’d for an event, or a student canceling an RSVP made by another user.

Club

The club entity represents a club in the system. A club will have student administrators, create funding applications and host events.

Event

This entity represents an event that is hosted by a club. An event is hosted in a venue at a specific date and time. The event has a title and description, detailing the event, as well as a maximum capacity. The capacity of the event should be less than or equal to the venue capacity it is hosted at. There is also an attribute ‘numRsvps’ to keep track of the number of RSVPs. The cost attribute is for the purposes of managing allocated club funds and is per the specification.

Venue

This entity represents the venue that an event will be hosted in. It has a maximum capacity, operating hours, cost, floor, room and description. We included floor and room attributes so each room from the same building can be its own specific venue. The room attribute will be implemented to allow for events occurring across multiple rooms. The description attribute will be used for miscellaneous information such as directions, dress codes, etc.

Funding Application

This entity represents a funding application by a club. The application can be reviewed by faculty members, with a status from the status enum.

Status

An enum that represents the different statuses that a funding application can be in (in draft, submitted, in review, approved, rejected).

Associations and Multiplicities

User ↔ Student and User ↔ Faculty

The User is a parent class with Student and Faculty Member as children, this accurately models that Students and Faculty are types of users and avoids duplicating attributes.

Student ↔ Club

We used a many-to-many multiplicity for this relationship, since clubs can have multiple student admins and a student may be an admin of multiple clubs. We are assuming here that every club must have at least one admin at all times.

Club ↔ Event

A club can host zero to many events as each club does not have to host an event. Each event only links to one club. Hence, it is a one to many relation.

Student ↔ RSVP

A student can RSVP for multiple attendees, but we decided to model RSVPs individually using the rsvpBy attribute (see RSVP entity description above). Therefore, it is a one to many relation, since a single student can have multiple RSVPs to different events but each RSVP is only for an individual student.

RSVP ↔ Event

Each RSVP is only linked to one event, hence the one to one relation.

Event ↔ Venue

A single event is hosted at one venue, hence the one to one relation.

Club ↔ FundingApplication

A club can apply for 0 to many funding applications, but each application is linked to a single club. This is a 1 to many relation.

Note: While a student club can only submit one application per semester, we are accounting for the storage of funding applications from past semesters, hence the many in the relation.

Faculty Member ↔ FundingApplication

A faculty member can review zero to many funding applications and each funding application may be reviewed by multiple faculty members, hence the many to many relation.