UPBoints

EVENTS BEYOND THE POSSIBLE

Join our Event!

UPBoints is an event management platform. In order to showcase its functionality, we've made Intersections an interactive event! Follow the QR code to win points, see your vouchers, and experience the platform for yourself.

upboints.netlify.app

James Bish
College of Arts and Sciences

Calvin Cai
Case School of Engineering

Annika Markoff
College of Arts and Sciences

Joshua Hager
Case School of Engineering

Charles Lin
Case School of Engineering

John McCormick
Case School of Engineering

Carson Williams
Case School of Engineering

Chelsea Yu Zheng
Case School of Engineering

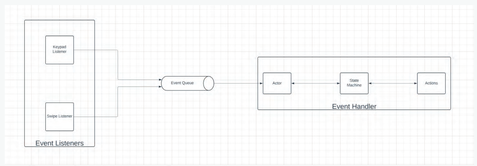
About

UPBoints is a project inspired by the need for an event management system to configure custom events for campus organizations. Organizations such as UPB (University Program Board) program events for up to 6,000 students, necessitating an efficient and secure system to track attendees and handle events. Our project is a web platform coupled with physical devices (terminals) to check in attendees, distribute vouchers and points for different activities, and track event statistics.

Goals

- Consolidate all features of a potential event on one web platform
- Increase the speed of checking in students to events
- Ensure that students are able to see earned vouchers, points, and raffle entries
- Make each feature customizable by event managers, ensuring platform longevity and flexibility in event creation
- Ensure that student data is securely handled
- Allow event terminals to function offline

Technical Details: Backend



Event Architecture: Several listeners monitor events that we are interested in, such as a keypad press or a swipe. These events are placed into an event queue, which is being emptied by the event handler.

Events are first given to the Actor. The Actor is responsible for taking in events and deciding what they are – i.e. is this a manager swipe, a user swipe, a keypad input, etc. The Actor then feeds the event into the State Machine, which performs the appropriate state transition depending on the current state as well as the input received. Any business logic needed on state transition is performed by the Actions class, as called upon by the state machine.

To facilitate the state machine, we have created a custom, lightweight state machine library called "pysm" to be used in this project.

To host, we used Netlify, which allows for the creation of serverless functions. These can be invoked without the need for an always-on server. Our serverless functions allow CampusGroups, Mongo Atlas, and Netlify to communicate, which help retrieve and update data from the end systems.

User Experience

UPBoints is a platform designed for three types of users: Event Attendees, Event Managers, and Event Administrators. Each status has distinct features and privileges, with administrator privileges having the greatest scope, and event attendees having the least.

Attendees

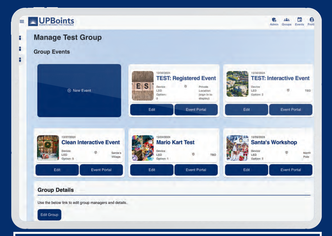
Attendee View of all upcoming events. This view is auto-populated by the upcoming events of CampusGroups organizations the attendee is part of.

The experience for an event attendee is customized by the event creator. Attendees have the ability to view the customized raffles, point distributions, and vouchers, as well as enter raffles directly on the web platform.

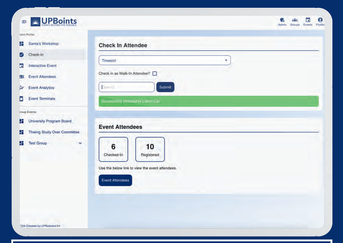
Interactive Event View, displaying an overview of available points and raffles

Interactive Event View, displaying earned points and available vouchers

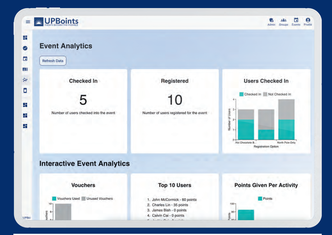
Event Managers



Overview of all events in a manager's group



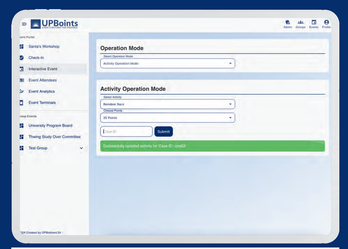
The check in attendee page. Check-in speed is now three times faster compared to using CampusGroups



The Event Analytics page allows managers a quick overview of event statistics

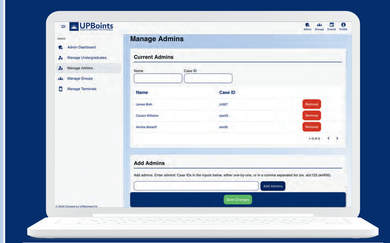
The bulk of the platform functionality belongs to managers. Managers have the ability to fully customize and operate events, ensuring a seamless experience for an event attendee.

Manager pages are generally optimized for desktop, with the assumption that desktop will be the most frequent mode of accessing the site with this status.

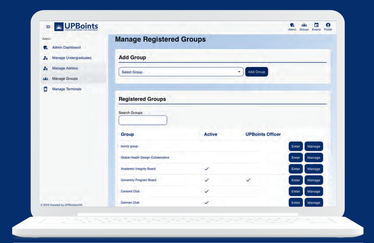


Any functionality that is available on the physical terminals has a web-portal counterpart, including operating activities and redeeming vouchers

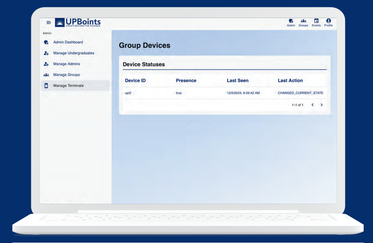
Platform Administrators



Admins alone have the ability to add and remove other admins



Editing the group involves modifying group permissions, adding managers, setting the group as active, and more



Having an overview of the devices allows administrators to track the terminals

Admins ensure longevity, as they can add new groups to the platform as well as create new admins

Devices

The physical terminals are built from Raspberry Pi Zero 2 Ws and programmed with Python. The devices have a local MongoDB database to store offline changes and update when a network connection is available, which allow offline functionality.

Devices are programmed to only work on the CaseRegistered network and require a manager or admin to swipe their ID card before being logged in to use a device. All actions on the device are logged and can be tracked from the web portal.

Technical Details: Frontend

The frontend leverages React as its primary library. The React project is bootstrapped via Vite, which allows for efficient development and production runtime. The entire project is then hosted publicly via Netlify, which has built in functionality to support Vite. The page's routing needs are handled via React Router, which allows the definition of dynamic routes for the various pages of the site. All styling is done by custom SCSS files, and the MaterialUI library is extended for many of the common inputs such as dropdowns, modals, tables, and moreparagraph text.

Testing

Netlify has built in CI/CD, which was used to test the frontend. Deploy previews meant that full-system functionality could be tested before integration to the main project, allowing testing of front end and back end integration. Additionally, the frontend team took advantage of Chrome Dev Tools to check responsiveness and device compatibility.

We have run pilots with the platform at 23 events this semester, and have recieved feedback and monitored for discrepancies or abnormalities in a live event scenario.

Future Work

The UPBoints Platform is currently set to be used on Sunday, December 8th, at the Thwing Study Over. As the platform begins to be used around campus, further features will likely be suggested, and the project will remain a living work for the remainder of the 2024-2025 school year.

Functionality has been planned to support the remote control of event terminals. Throughout testing at events, we have received feedback regarding the learning of device features without a screen on the device. We plan to allow devices to be controlled from the website, helping users set up the devices and learn the functionality.