Projects

List of projects the Tech Team has worked on

- Blueprint Admin - Spring 2024
- Kudos Design Document
- Staging Environment
- Kubernetes
- Blueprint Admin Design Spec

Blueprint Admin - Spring 2024

Blueprint Admin Spring 2024

We are glad that you have decided to join the Tech Team this semester. We hope this experience serves as an introduction to the world of software development. By the end of the semester, you will be able to improve your coding skills and have a greater understanding of Web/API development.

Overview of the project

To help our Project Teams develop software faster, we provide the teams with a staging environment. Here, Project Teams are able to deploy a production-ready application that will help them test new features and showcase their progress to the NPOs. We don't want anyone to have access to this staging environment. Therefore, we use a tool named SSO (Authelia).

Have you noticed that whenever you try to access Canvas or Workday, you are prompted to log in to a page? This is an SSO. It is a way for Blueprint to have a homogenous login. The SSO we use is called Authelia. The way Authelia retrieves the users with permission is through a YAML file. For example:

```
users:
  user1:
    disabled: false
    displayname: Blueprint User 1
    password: existingpassword
    email: user1@blueprint.com
    groups:
      - admin
      - dev
  user2:
    disabled: true
    displayname: Blueprint User 2
```

```
    password: existingpassword
    email: user2@blueprint.com
    groups:
      - admin
```

The main feature of our project is having a way to manage this YAML file. We need to be able to add, delete, disable, and update users in this YAML file. However, we also want to extend the functionality of our application by adding new features such as Team Management, Finances, Blog management, and Event Management.

Blueprint Admin (Frontend)

The frontend application is currently being developed with React using TypeScript. The web application should have the following pages:

1. Member Management
2. Application Management
3. Team Management
4. Budget Management

Member Management

The User Management page serves as the central hub for administrators to oversee and control user access and profiles within the system. This page allows for creating, editing, and deleting user accounts, enabling administrators to assign or revoke permissions and access levels. Key features include user search, filter options to locate users quickly, and detailed user profile views that display login history, activity logs, and personal settings. Through this interface, administrators can also reset passwords, manage roles (e.g., admin, user, guest), and set up multi-factor authentication to enhance security.

Application Management

The Application Management page is tailored to oversee and process applications submitted to the organization. This platform serves as a centralized system for administrators to review, sort, and respond to various applications efficiently. Key functionalities include viewing and assessing each application, tracking its status (e.g., received, under review, approved, rejected), and managing communications with applicants directly from the interface.

Team Management

The Team Management page focuses on facilitating team organization, collaboration, and productivity within the company or system. It provides tools for creating and managing team structures, including working groups. Administrators can assign members to teams, set roles and responsibilities, and track progress on team objectives or projects. Features may include shared

calendars, task assignments, performance metrics, and communication tools to support effective teamwork. The page aims to centralize team resources and information, making it easier to manage workflows and ensure that team members are aligned with their goals and deadlines.

Budget Management

The Budget Management page provides a comprehensive way to track Blueprint's budget. Throughout the semester, we organize various events for which we need a budget. This page will help us track how much of our budget we have allocated to events and how much we have left for future events. Since we are migrating our servers to a cloud provider, we need to see if our server usage will exceed the budget.

Blueprint Admin (Backend)

User Management

Kudos Design Document

Problem

As students and developers, our lives move fast, and accomplishments for our work often goes unnoticed. Since we are tasked with executing and delivering software on top of schoolwork and other responsibilities, it is easy to feel overworked and burnt out.

The Tech Team will create a tool that facilitates interconnectivity across project teams by allowing developers to leave feedback under each other's work during sprints.

Feedback can be positive or constructive, and will always be anonymous.

Features

The app consists of two main pages.

Features for page 1, the main page:

- a 4 x 4 card layout, where each card contains the following:
  - commit content headline
  - author
  - streak/trend info (if applicable)
  - a text input box
- a sidebar that allows users to navigate team archive. It has:
  - teams dropdown -> clickable sections to view the commits for each project team
    - team 1
    - team 2
    - team 3
  - archive dropdown -> clickable archives with commits and kudos from past sprints
    - 3/21 sprint
    - 3/14 sprint
    - 3/7 sprint 1

each of these components will have filter, sort, and search functionality

---

Features of page 2, the admin page:

Page 2 will have admin access only.

- dashboard layout where an admin can do the following:
  - **start a kudos session**. admins can open up the kudos board for developers to visit and interact with. the vidibility of the board changes at the admin's discretion.
  - **review all comments**. admin can see comments on all commits (with username attached?) (identify user who made comment, but dont display content?)
  - **view comment analytics**. admin can see how many comments a developer has made by sprint or all time.
  - **set timed start and end**. admin can set a timer that will be displayed on the maoin page indicating when the next session will be. once a session is in progress, another timer will be dispalyed indicating when the session will close.

End-to-End User Workflow

1. General User (Developer) Experience

- Initial Access:

- The user opens the application in their web browser.
- The frontend makes an API call to the backend to fetch the latest commit data and active session status.
- The main page (4x4 card layout) is rendered, displaying commit headlines, authors, and any existing streak/trend information.
- The sidebar is populated with team and archive navigation options.

- Browsing Commits:

- The user can scroll through the 4x4 card layout to view different commits.
- They can use the sidebar to navigate to specific teams or archived sprints.
- Filtering, sorting, and searching functionalities allow them to find specific commits.
- The frontend send the filter, sort, or search parameters to the backend via API calls, and the backend returns the filtered, sorted, or searched data.

- Leaving Feedback (Kudos):

- If a kudos session is active (indicated by a timer or clear visual), the user can enter feedback in the text input box within each commit card.

- When the user submits feedback, the frontend sends a POST request to the backend API, including the commit ID, user ID (anon), and the comment content.
- The backend stores the comment in the database, associating it with the commenter, the commit, and the author.
- (If using websockets, the comment is broadcasted to other users in real-time???)

- Viewing Session Status:

- The user can see the status of the current kudos session (active, inactive, time remaining) on the main page.
- This information is fetched from the backend and updated in real time if web sockets are implemented.

2. Admin User Experience

- Accessing Admin Page:

- The admin user logs in with their credentials.
- The frontend checks the user's role and redirects them to the admin page if authorized.

- Starting/Ending a Kudos Session:

- The admin uses the dashboard to start or end a kudos session.
- The frontend sends a POST request to the backend API to update the session status in the database.
- The backend updates the database and, if using WebSockets, broadcasts the session status change to all connected users.

- Reviewing Comments:

- The admin can view all comments on the dashboard.
- The frontend sends a GET request to the backend API to retrieve all comments.
- The backend retrieves the comments from the database and returns them to the frontend, possibly with user identification (for admin review only, not public display).

- Viewing Comment Analytics:

- The admin can view comment analytics, such as the number of comments per developer and per sprint.
- The frontend sends a GET request to the backend API to retrieve the analytics data.
- The backend performs the necessary data aggregation and returns the results to the frontend.

- Setting Timed Start/End:

- The admin can set the start and end time of a kudos session.
- The frontend sends the start and end times to the backend via a POST or PUT request.
- The backend stores the times in the database and uses them to control session status and display timers on the main page.
- If web sockets are used, the time remaining will be updated in real time.

High Level Architecture

1. Frontend (Client-Side):
   - React/TSX UI
     - handling user interactions, and displaying data.
     - rendering the 4x4 card layout, sidebar navigation, and admin dashboard.
     - API calls to backend db.
2. Backend (Server-Side):
   - Django/Python for the APIs.
     - manages data retrieval from GitHub API
     - handles auth using GitHub API
     - implements logic for filtering, sorting, and searching commits and comments.
3. Database???:
   - MongoDB to store commit data, user information, comments, and session details.
4. Real-time Communication???:
   - websockets?

Outstanding Questions

should we store commit data? or just pull instances from API?

Staging Environment

The blueprint staging environment serves two primary purposes.

- Hosting the staging environments for our project teams
- Hosting the infrastructure powering blueprint internal operations

Currently the staging environment is hosted on one singular machine sponsored by EzriCloud. As Stevens IT is unable to provide us with a working cloud machine. In the future, we hope to move our infrastructure to AWS once we have a stable funding for it.

Ezri Zhu should be the primary contact for any server issues. They can be reached via their discord.

The server's NixOS configuration code is made available here

https://github.com/stevensblueprint/techops/

Below are the things that is currently on stag0.nyc.sitblueprint.com

NYCMesh team staging service (deployed via docker compose)

AAD-ADMIN team staging service (deployed via docker compose)

C4P team staging service (deployed via docker compose)

Authelia SSO w/ nginx reverse proxy (deployed via NixOS configuration)

Vaultwarden password manager (deployed via NixOS configuration)

Bookstack (this wiki) (deployed via NixOS configuration)

Blueprint internal admin dashboard (deployed via docker compose)

Blueprint internal admin backend (deployed via docker compose)

Blueprint internal user management service (deployed via docker compose)

Kubernetes

For the fall 2024 semester, we are planning on launching our internal k8s platform to run the things that we're currently running on docker compose.

Although using kubernetes at our current scale is probably not the greatest idea in terms of maintainability, we are still choosing to do this so that interested tech team members can learn to operate k8s containers.

We will likely be using k3s - https://k3s.io/

Ezri Zhu will be the project owner for the k8s cluster, they can be reached via discord.

This page is currently WIP.

Blueprint Admin Design Spec

Blueprint Admin Design Spec

Problem Overview

We develop software for non-profits at no cost. To promote talent at Blueprint we run the Tech Team that serves as low commitment team where Stevens students can come and learn software engineering skills. Currently, Blueprint has around 40 active members that work on the projects with the NPOs and in the Tech Team. Software development and the usage of computing and building tools. Managing the access of said resources for every member has become unsustainable for the Blueprint e-board and Tech Lead for the project Teams. Having a centralized solution to administer Blueprint resources and manage member is necessary for the future success of the organization.

Solution Overview

To mitigate the need for a management solution, the Tech Team has started building a user/resource management solution. Most of the backend service that will power the admin has already concluded, however designs for the front-end have to be formalized.

Feature Description

The admin dashboard should provide the following features:

- Be able to log-in as an e-board, Teach Lead. Note: A single member might belong to multiple roles, so when the member logs in they should have the feature available for each of the roles they belong to.
- Once logged in, the member should have a sidebar where they can edit their information, such as name and password. A member should also have profile photo (Mostly all member will have a default profile photo, it could be the blueprint logo, or a stock user photo)

E-board functionality

Blueprint E-board members should be able to do the following:

- Send an invitation email to an interested member given the email and name of the member who wants to join

- View a table of all the members
  - The displayed information of the members is the following:
    - Roles (List of Strings)
    - Team they belong to (List of String)
    - Date joined (LocalDate)
    - Active (Boolean)
  - The table can be filtered by the afore mentioned columns
  - The e-board member can edit each of the field of a member

Team Lead functionality

Blueprint Team Leads should be able to do the following:

- View a list of all the member associated to their team
- Edit the active status of a member team
- Take attendance of each of the meeting
  - The Team Lead should be able to create an attendance record with some basic information such as Date and Location
  - The Team Lead can only take attendance on the active members of the team
- View statistic on the attendance of the Team such as:
  - Drop out rate of the team (How many people have been deactivated since the start of the project?)
  - Attendance rate