# Team Contributions: POC Sandlot

Team 29
Nicholas Fabugais-Inaba
Casra Ghazanfari
Alex Verity
Jung Woo Lee

This document summarizes the contributions of each team member up to the POC Demo. The time period of interest is the time between the beginning of the term and the POC demo.

## 1   Demo Plans

To be a successful project, the project should be able to intake a set of availability data and generate an optimized schedule based on that data. Additionally, it should be able to display this schedule through a basic user interface. The main risks for the success of this project are whether we are able to optimize schedule generation to a point which satisfies our stakeholders, and our team's lack of domain knowledge on scheduling problems and web development.

In order to demonstrate that these obstacles can be overcome, the goal of this proof of concept demonstration is to develop an algorithm to generate highly optimized schedules based on team availability data. Furthermore these optimized schedules should be displayed through a basic UI hosted on a webvserver.
Demonstration Plan:

- Quick overview of the POC demonstration

    Scheduling algorithm and UI visualizer

- Explanation of scheduling algorithm

    Problem formulation

    Algorithm to solve

    Algorithm details and other notes

- Demonstration of POC

    Show UI

Generate schedule

Show that each constraint is satisfied

## 2 Team Meeting Attendance

| Student | Meetings |
|---|---|
| Total | 5 |
| Nicholas Fabugais-Inaba | 4 |
| Alex Verity | 4 |
| Jung Woo Lee | 5 |
| Casra Ghazanfari | 5 |

The total for the number of meetings attended could be seen as being understated for all members. This is because the team has had many minor impromptu meetings both online in person which did not have specific agendas or were simply to complete more work without discussion. These minor meetings were not deemed significant enough for team meeting issues to be created for them.

## 3 Supervisor/Stakeholder Meeting Attendance

| Student | Meetings |
|---|---|
| Total | 4 |
| Nicholas Fabugais-Inaba | 4 |
| Alex Verity | 4 |
| Jung Woo Lee | 4 |
| Casra Ghazanfari | 4 |

## 4 Lecture Attendance

| Student | Lectures |
|---|---|
| Total | 9 |
| Nicholas Fabugais-Inaba | 9 |
| Alex Verity | 7 |
| Jung Woo Lee | 8 |
| Casra Ghazanfari | 8 |

# 5   TA Document Discussion Attendance

| Student | Meetings |
| --- | --- |
| Total | 3 |
| Nicholas Fabugais-Inaba | 3 |
| Alex Verity | 3 |
| Jung Woo Lee | 3 |
| Casra Ghazanfari | 3 |

# 6   Commits

| Student | Commits | Percent |
| --- | --- | --- |
| Total | 236 | 100% |
| Nicholas Fabugais-Inaba | 48 | 20% |
| Alex Verity | 85 | 36% |
| Jung Woo Lee | 59 | 25% |
| Casra Ghazanfari | 44 | 19% |

Alex's commits are higher as we have worked off his computer together as a team.

# 7   Issue Tracker

| Student | Authored (O+C) | Assigned (C only) |
| --- | --- | --- |
| Nicholas Fabugais-Inaba | 39 | 14 |
| Alex Verity | 0 | 15 |
| Jung Woo Lee | 10 | 13 |
| Casra Ghazanfari | 22 | 14 |

Nicholas has created all the issues for the deliverable documents and Casra has created most TA Feedback issues.

# 8   CICD

- Continuous Integration

Continuous integration will be implemented via GitHub Actions. When a pull request is made, automated tests will run against the code being tested.

- Unit testing frameworks

The pytest framework will be used to create unit tests for the middleware code. We chose pytest over other python testing frameworks due to its simplicity, small amount of boilerplate code, and plugins which can add useful functionalities like coverage reporting. We plan to incorporate these pytest unit tests as a part of our CI plans for the project via Github actions.

Testing the database will likely be done using a dummy / development PostgreSQL database prior to making any changes to the production database to ensure that minimal migrations are required during development.

- Code coverage measuring tools

The Coverage.py Python library will be used to measure the code coverage of our middleware program. For the webserver's React code, Jest is included by default when using the "create-react-app" command and will be used to measure the code coverage of the webserver.