System and Software Architecture Description (SSAD)

Field Progress Web App

Team Number 04

Uche Uba – Project Manager

Madhavi Shantharam – Life Cycle Planner

Akanksha Diwedy – Operational Concept Engineer

Sahithi Reddy Velma – Software Architect

Aishwarya Joisa – Feasibility Analyst

Mayank Kulkarni – Requirements Engineer

Kevin Grimes – IV&V

28-10-2019

Version History

Date	***Author***	***Version***	***Changes made***	***Rationale***
10/28/19	SV	1.0	• Added all the required sections for SSAD document.	• Final draft required for the DC package.

Table of Contents

Table of Tables

Table of Figures

1. Introduction

1.1 Purpose of the SSAD

The purpose of the SSAD is to provide an architectural view of the project. It provides the details about the system architecture as well as the components that it is composed of. The document can also be used to identify the main actors of Field Progress Application and how they would interact with system.

1.2 Status of the SSAD

This document is the first draft of the SSAD and thus there are no differences from a previous version.

2. System Analysis

2.1 System Analysis Overview

The primary purpose of Field Progress App is to act as a means, through the use of technology to help campaign managers travel to their voters more efficiently. The aim of the project is to automate the process of turf cutting for the campaign managers. This way we free up time for the campaign managers to put towards other productive activities. The application aims to take various parameters about volunteer availability as input and provides a visualization of the most efficient route on a map.

2.1.1 System Context

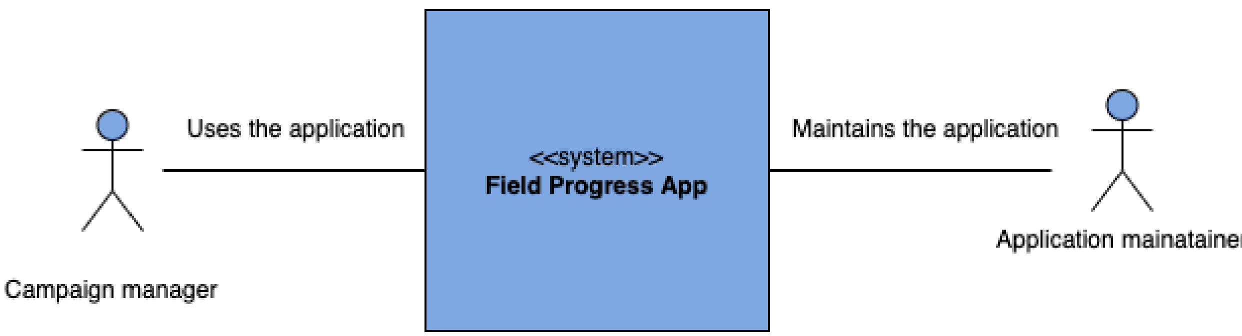

Figure 1: System Context Diagram

Table 1: Actors Summary

Actor	Description	Responsibilities
Campaign Manager	A user who takes up the responsibility of overlooking campaigning activities for a candidate	• Assign volunteers to travel to the voters in a district. • Cut out Turfs based on volunteer availability.
Application Maintainer	A person who maintains the upkeep of the application	• Fixes bugs in the algorithm • Updates the application with voter data from upstream.

2.1.2 Artifacts & Information

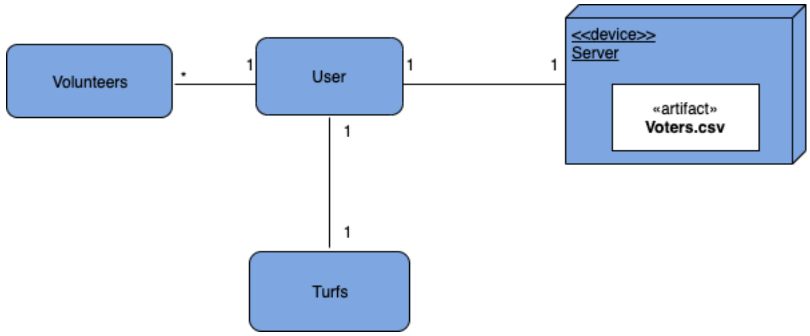

Figure 2: Artifacts and Information Diagram

Table 2: Artifacts and Information Summary

Artifact	Purpose
ATF-1: Voters	Contains voter data in a csv format which would be an input to the turf-cutting algorithm
ATF-2: Volunteers	Contains information about volunteers like name and time of availability which would be inputs to the algorithm
ATF-3: Turfs	Contains information about the cluster each voter belongs to along with the volunteer assigned to that cluster. This is the result of the turf-cutting algorithm.

2.1.3 Behavior

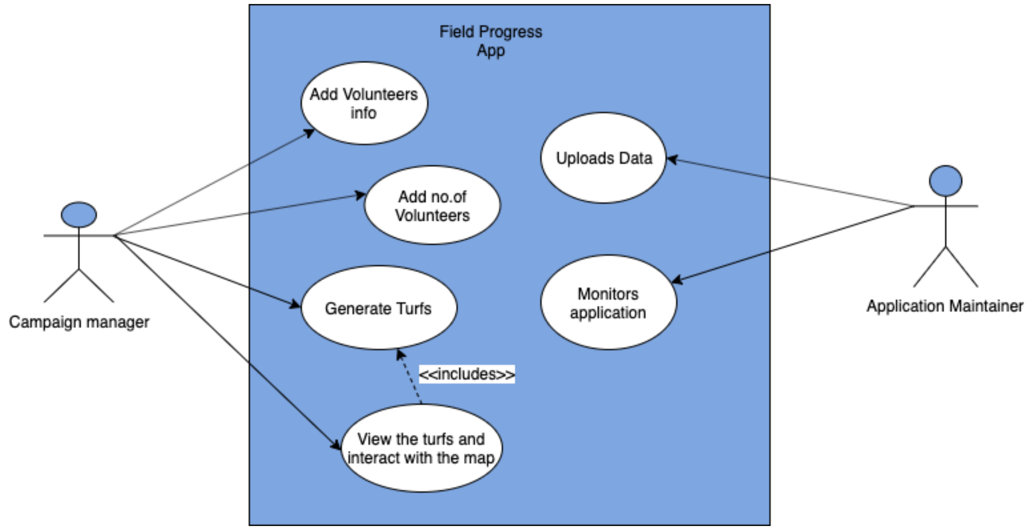

Figure 3: Usecase Diagram

2.1.3.1 Input Volunteer Parameters

2.1.3.2 Add Volunteers Information

Table 3: Process Description- Add volunteers' info

Identifier	UC-1 Add volunteers Info
Purpose	Gives volunteer availability times as input to the algorithm
Requirements	WC_5508
Development Risks	Taking into account availability time can be challenging to implement as part of the algorithm.
Pre-conditions	User is currently viewing the turf cutting form.
Post-conditions	After entering the volunteer info, the algorithm would take into account these parameters and cut turfs accordingly.

Table 4: Typical Course of Action – Add volunteers' info

Seq#	Actor's Action	System's Response
1	Click on add volunteer button	
2	Enter volunteer info in the form generated	
3		Validate input.

Table 5: Alternate Course of Action – Invalid Inputs

Seq#	Actor's Action	System's Response
1-3	Refer to typical course of action	
4		Clears the text box for volunteer name and availability time.
5		Displays an error message "Please check your inputs" below the add volunteer form.

2.1.3.1.2 Add no. of Volunteers

Table 6 – Process Description – Add no. of Volunteers

Identifier	UC-2 Add no. of volunteers
Purpose	Gives no. of volunteers as an input to the clustering algorithm
Requirements	WC_5508
Development Risks	Generating turfs taking into account the no. of volunteers available can be challenging.
Pre-conditions	User is currently viewing the turf cutting form.
Post-conditions	After entering the volunteer info, the algorithm would take into account these parameters and cut turfs accordingly.

Table 7: Typical Course of Action – Add no. of volunteers

Seq#	Actor's Action	System's Response
1	Click on add no. of volunteers button	
2	Enter no. of volunteers in the input field.	
3		Validate input.

2.1.3.2 Turf-Cutting

2.1.3.2.1. Generate Turfs

Table 8: Process Description- Generate Turfs

Identifier	UC-3 Generate Turfs
Purpose	Generates turfs for assigning volunteers to a turf.
Requirements	WC_5696
Development Risks	None.
Pre-conditions	User is currently viewing the turf cutting form User has entered the volunteer info and no. of volunteer's parameters.
Post-conditions	User will be to view the cut-out turfs on a map.

Table 9: Typical Course of Action- Generate turfs

Seq#	Actor's Action	System's Response
1	Click on Cut Turfs button.	
2		Create a list of voters and assigned cluster.

2.1.3.1.2 View Turfs

Table 10: Process Description- View Turfs

Identifier	UC-4 View Turfs
Purpose	Visualizing the cut-out turfs on a map.
Requirements	WC_5696
Development Risks	None.
Pre-conditions	User is currently viewing the turf cutting form User has entered the volunteer info and no. of volunteers

	parameters.
Post-conditions	User will be to view the cut-out turfs on a map.

Table 11: Typical Course of Action- View Turfs

Seq#	Actor's Action	System's Response
1	Click on View Turfs button.	
2		Create a Visualization of turfs on a map.

2.1.4 Modes of Operation

Field Progress application will operate only in one mode; therefore no description is stated in this section.

2.2 System Analysis Rationale

The Field progress app is an application that allows campaign managers to travel to their voters more efficiently by automating the process of turf cutting and picking out the most efficient route for the volunteers.
The main users of the application are the campaign manager along with a maintainer.
Campaign Manager –
This is a user who is entrusted with the responsibility of overlooking the campaigning activities of a candidate. The campaign manager can use the field progress app to get auto cut -out turfs to assign volunteers effectively and also provide an optimal route so that the volunteers can reach their potential voters in a time efficient way.
The campaign manager enters the volunteer's info i.e. volunteer name and time of availability in the form and can view the visualization of the turfs on map and can get a list of volunteers assigned to a turf based on the inputs.

3. Technology-Independent Model

This section has been omitted as we have already decided on the technology that will be used. Therefore, to avoid redundancy, the diagrams in this section have been specified in the next section.

4. Technology-Specific System Design

4.1 Design Overview

4.1.1 System Structure

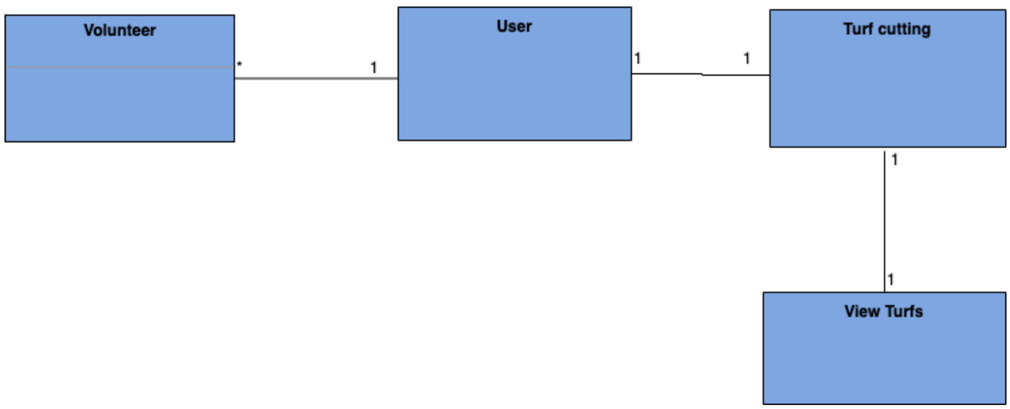

Figure 4: Conceptual Domain Model

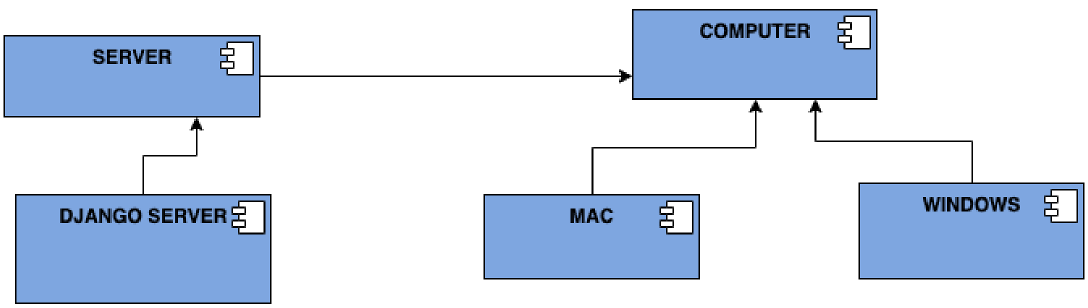

Figure 5: Hardware Component Class Diagram

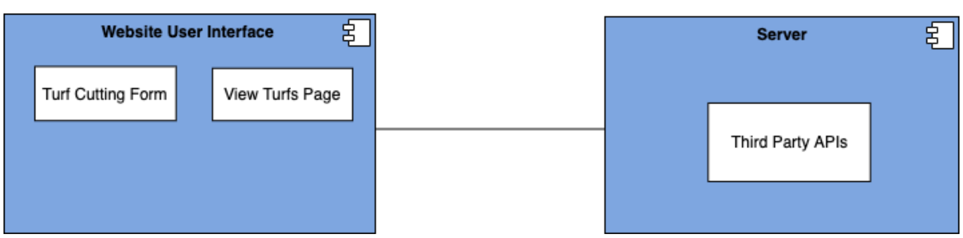

Figure 6: Software Component Class Diagram

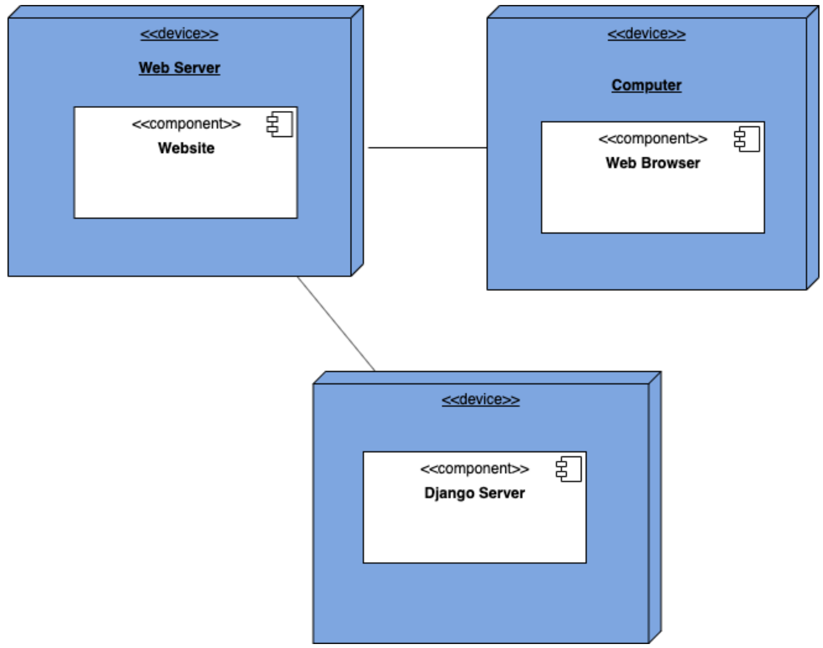

Figure 7: Deployment Diagram

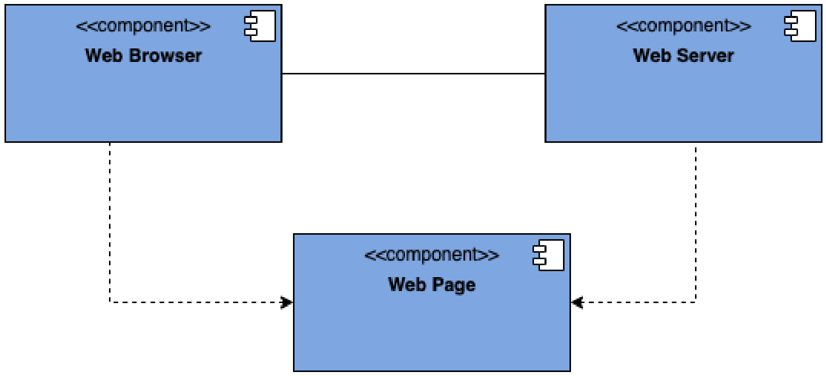

Figure 8: Supporting Software Component Class Diagram

Table 12: Hardware Component Description

Hardware Component	Description
Server	The web server is used for hosting the application and for the Django server. All the API calls to the algorithm (generate turfs) are sent to the web server.
Computer	Web application will be accessed from a computer, which can be either Windows or MacOS.

Table 13: Software Component Description

Software Component	Description
User Interface Component	This component contains Field Progress Application web pages or use by all users
Turf Cutting Form	This page allows the user to input various parameters of the volunteers in the form as an input to the algorithm
View Turfs Page	This page allows the user to view and interact with the turfs displayed on a map.
Third Party APIs	APIs provided by third party like google maps APIs for implementing features like walkability.

Table 14: Supporting Software Component Description

Support Software Component	Description
Browser	An internet browser that connects to the Field Progress web application and displays the web pages.
Web Server	The server component that deploys the application.
Web Pages	The web pages part of the field progress app.

4.1.2 Design Classes

4.1.2.1 System Diagram

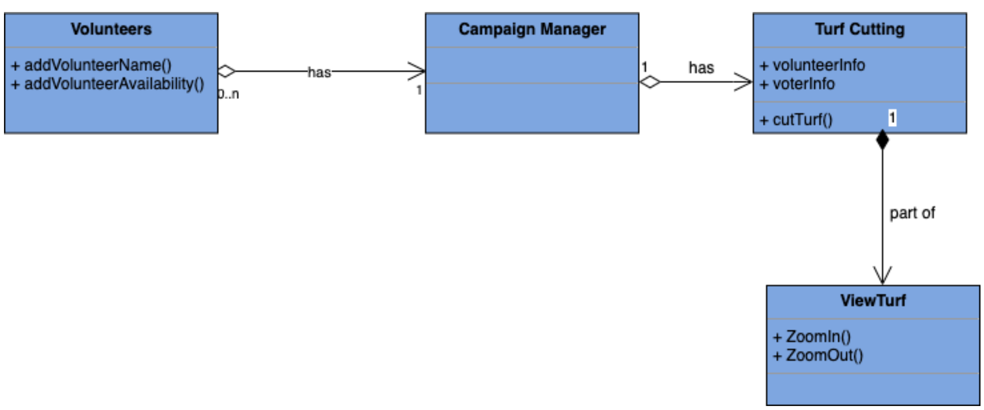

Figure 9: Design Class Diagram

Table 15: Design Class Description

Class	Type	Description
Campaign Manager	Entity	User who uses Field Progress App for help with campaigning responsibilities.
Volunteer	Entity	Contains volunteer data like name and time of availability.
Turf cutting	Component	Component for cutting out turf given volunteer and voter info.
View	Component	Section for visualizing the cut-out turfs on map.

4.1.3 Process Realization

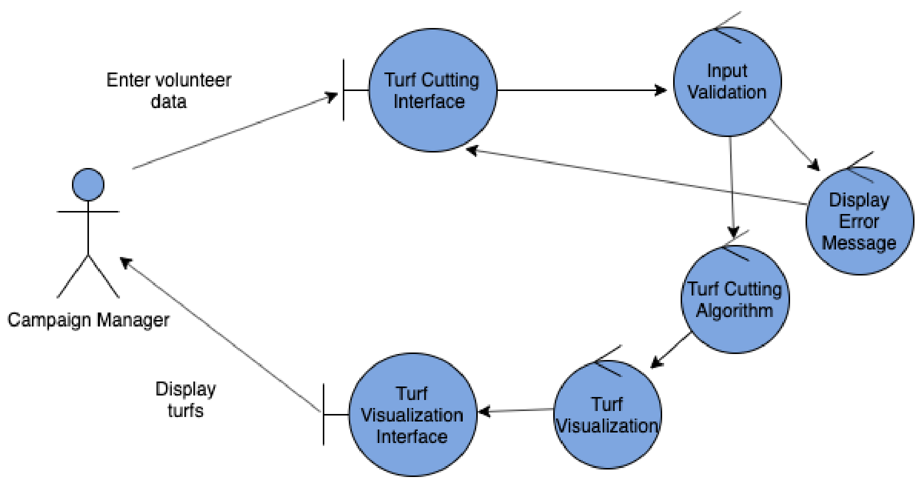

Figure 10: Robustness Diagram

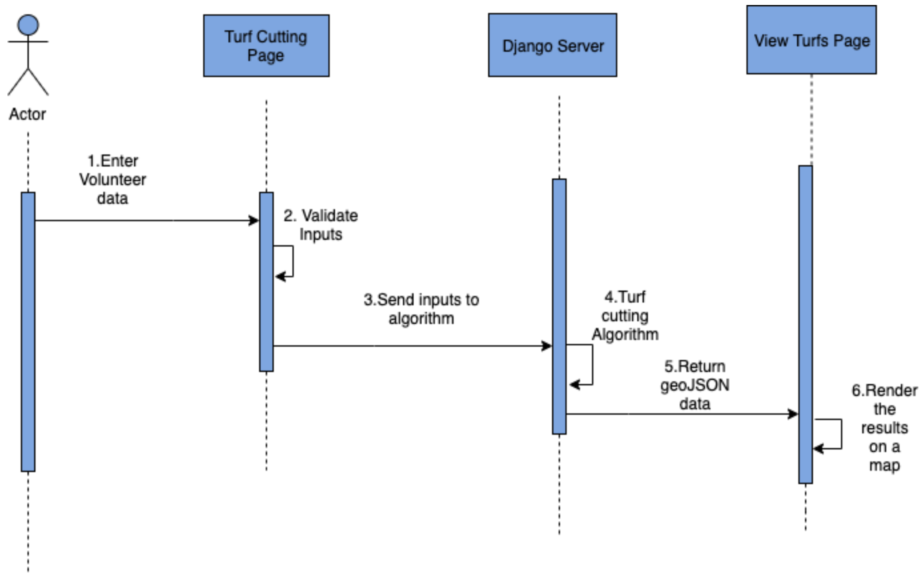

Figure 11: Process Realization Diagram

4.2 Design Rationale

This is a web application with the focus being on the efficiency of the algorithm for cutting- out turf. Along with the algorithm, we have also provided a visualization in order to view the results of the algorithm.

To develop the front-end of our system, we used React.js because it is easy to learn, and component based. There is a time constraint, since it is a one semester project hence it is important to go with a framework that is easy to learn. Additionally, component based front-end helps in encapsulation and separation of concerns and hence makes it easier to develop as a team. We have used a Django server as the backend, since the algorithm was developed in python using libraries for implementing clustering, we decided to choose Django framework as this integrates well with python. All the HTTP requests would be routed to the url containing the algorithm. By having a back-end separate from the front-end we were able to achieve modularity in the application. Also, it makes it easier to make any changes in the future due to the separation of the two components.

5. Architectural Styles, Patterns and Frameworks

Table 16: Architectural Styles, Patterns, and Frameworks

Name	Description	Benefits, Costs, and Limitations
React.js	A JavaScript library for building reusable UI components for data that changes frequently.	Benefits: • Easy to learn • With reusable components time for development is reduced • With components, working in teams becomes easier. • Improved website performance with virtual DOM. Limitations: • Not all browsers support React.js
Django	A high-level web-based python framework.	Benefits: • Django framework takes care of the hassle of the web development, so that we could focus on the development of the algorithm. • Supports MVC pattern to keep UI and business logic separate. Limitations: • Makes web application components tightly coupled.