SDSU | San Diego State University

Kili Trekker Software Design Specification

By: Neel Raj, Matteo Gristina, Ryan Jaber

Prepared by: Bryan Donyanavard

Table of Contents

System Overview……...2

Software Architecture……...3

1. Architecture Description…….….3

2. UML Class Diagram…….….....4

3. Description of Classes…….4-5

4. Description of Attributes……5-6

5. Description of Operations…….….6-7

6. Development Plan and Timeline……...8-9

Base Test Cases…….……9

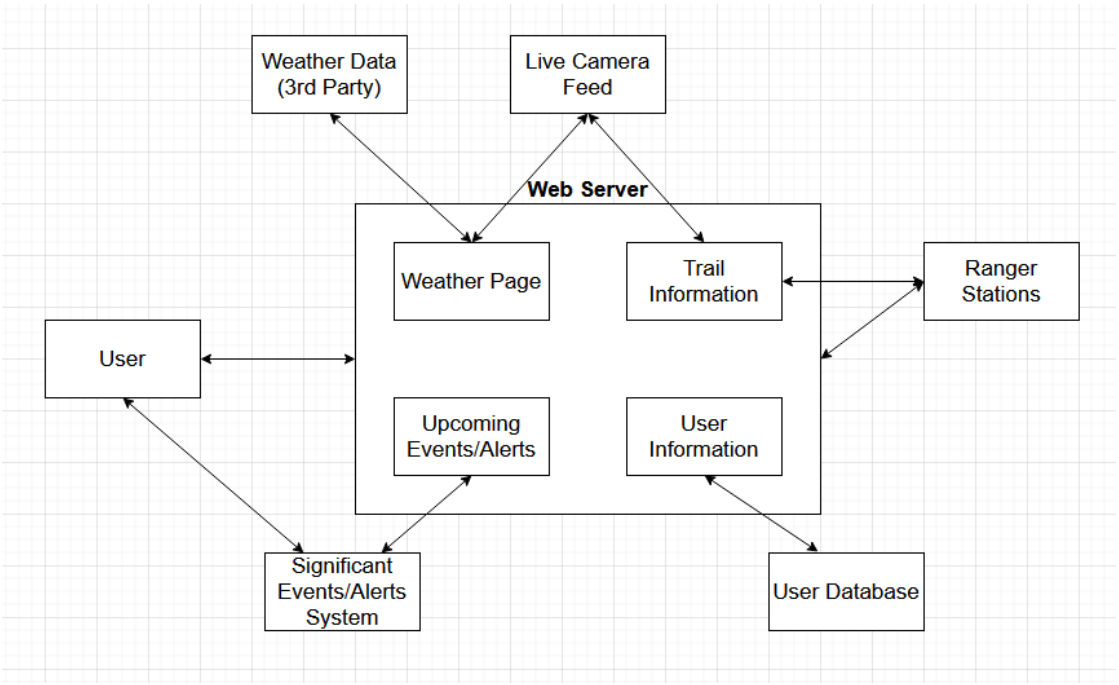
Team Responsibilities…….……9

System Overview:

The Kili Trekker System stores and displays information about and descriptions of the 12 trails leading to the summit, trusted hiking guide companies, and trekkers who have signed up with the guide companies. To help tourists and guides, the system will maintain weather data covering Kilimanjaro and the surrounding park, and features a camera at the top of the mountain that streams a live feed of the conditions. In addition, dates of upcoming events, local celebrations, and information regarding any park facilities under construction will be hosted and displayed to users. Trekker guides and rangers are able to update all the information in the system with their cellphones. The system is limited by a lack of cell coverage within the park and is also different cell technology than that of the U.S. Anyone that is not staff can only view information that is already displayed. The system will use existing infrastructure to link all ranger stations to the PC-based main server.

Software Architecture Overview:

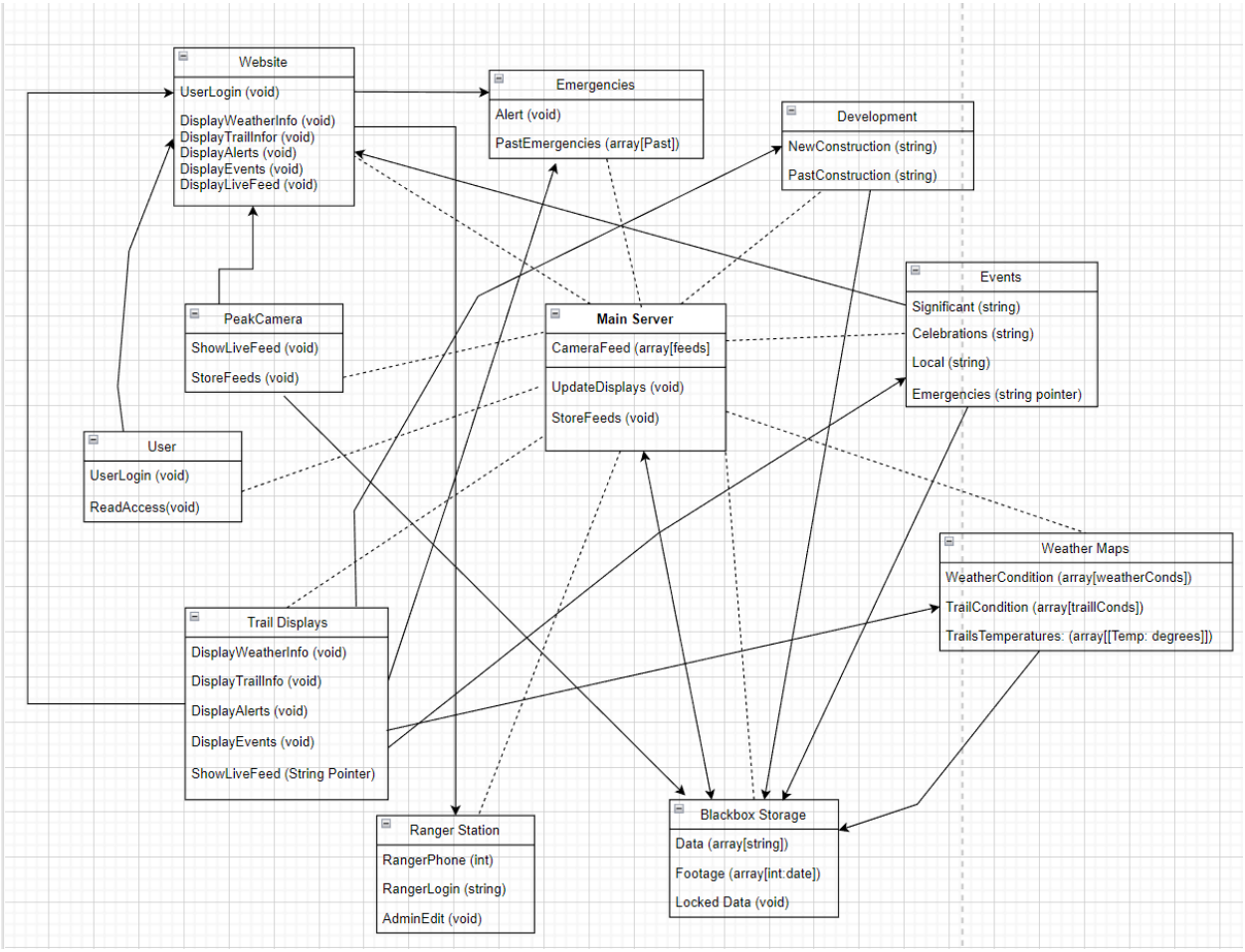
1. **Software Architecture**



Architecture Description:

The architecture for the Kili Trekker system is based on a simple web server/website, so it has a Client-Server Architecture. The user interacts with the main web server which is made up of 4 components: Weather, Trail Information, Upcoming Events, and User Information. The web server is accessible by the user in read format, the user cannot update any information. The weather page interacts in a two way fashion with 3rd party weather data and the live camera feed from the summit to accurately display the current and upcoming weather conditions. The server sends requests to both components and is sent back data to be displayed. The trail information cannot be automatically updated, so computers at ranger stations can be used to directly update information and trail conditions. The stations also have write access to the web system as a whole, as requested in the information given by the client. To keep track of users, the User information component reads and writes to an external database. Lastly, the upcoming events works together with the Alerts system to directly communicate with the User in case of emergency, or display non-emergency events occuring in the park. The User and Alert system form a Publish-Subscribe architecture, where those opted in to emergency notifications will receive them.

2. UML Class Diagram



3. Description of Classes

- *Website*: The main way users can interact with the info in the system.
- *Emergencies*: Updates emergency alerts for the system.
- *Development*: Updates new and old construction for users to see.
- *Events*: Updates any celebrations and local events for users to see.
- *Weather Maps*: Updates current weather conditions at trailheads.
- *Black Box Storage*: Backup Storage of all data collected as a safeguard, also links to the main server for rebel protection.

- *Ranger Station*: Lets rangers have read and write access to all info in the system.
- *Trail Displays*: Show all relevant info on displays located at trailheads.
- *User*: Lets users log in to the website and read all relevant info.
- *Peak Camera*: Shows live video feed of the mountaintop and stores feed.
- *Main Server*: Stores all info collected by other classes.

4. Description of Attributes

- *NewConstruction: string*- Displays new construction partaking on the grounds including dates as string cases.
- *Past Construction: string*- Displays past history of construction partaking on the grounds including dates as string cases.
- *Significant: string*- Displays significant events partaking on the grounds with dates as string cases.
- *Celebrations: string*- Displays Celebration events partaking on the grounds with dates as string cases.
- *Local: string*- Displays local events partaking on the grounds with dates as string cases.
- *RangerPhone: integer*- Displays ranger's phone number as an integer of 8 digits.
- *RangerLogin: string*- Contains ranger login information associated as a string, numbers are allowed.
- *WeatherCondition: array[WeatherCond]*- Set of weather conditions of the park and weather conditions of trails in an array filled with data as the data increases so should the indices.
- *TrailCondition:array[trailCond]*-Set of trail conditions of the park and its trails in an array filled with data as the data increases so should the indices.

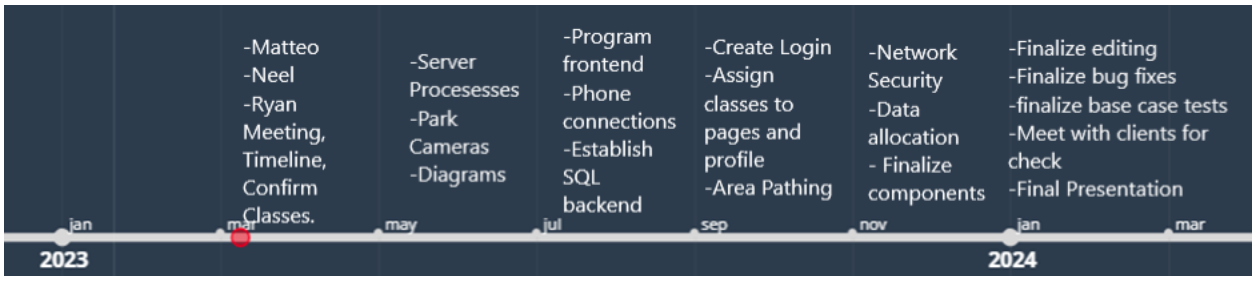
- *TrailsTemperatures: array[Temp: degrees]*- Set of Temperatures in degrees of the park and its trails in an array filled with data as the data increases so should the indices.
- *Data: array[string]*- All the data stored in the array should be increased with every new data index input. Depending on how many forms of data is colluding, 2D or 3D arrays may be used.
- *Footage:array[int:date]*- A 2D array using dates as indice holder and incorporates footage pixels until out of space. Make sure to add 5 years worth of data/RAM hardware till replacement is needed.
- *CameraFeed: array[feeds]*- Uses a 3D array to incorporate the feed's dates, Cam, and time when the feed was allocated when displayed.
- *Past Emergencies: array[Past]*- Uses a 2D array to form dates as indice placeholders and the footage being connected but only from the past emergencies that have been cleared.

5. Description of Operations:

- *DisplayWeatherInfo: void* - When called, displays all current weather info on the mountain.
- *UserLogin: void* - When called, lets user log in to the website to read all relevant info
- *DisplayTrailInfo: void* - When called, displays info about the trail on the trailhead displays.
- *DisplayAlerts: void*- When called, displays all alerts and emergency info for users to see. Called by rangers and staff.
- *DisplayEvents: void*- When called, displays all info about events and local celebrations on the website for users to see.

- *DisplayLiveFeed: void*- When called, displays live camera feed of mountain peak. Gets called on the website.
- *ShowLiveFeed: void*- When called, retrieves the live feed from the mountain peak camera. Gets called by the PeakCamera class.
- *StoreFeeds: void*- When called, stores all the feeds collected by the peak camera for the past month. Gets called by the main server where all data is stored.
- *ReadAccess: void*- When clicked it will only be read access.
- *AdminEdit: void*- When clicked, admin edit is enabled which allows changes only from ranger login.
- *Alert: void*- When called, pushes a new alert to the top of the stack of alerts.
- *UpdateDisplays: void*- When called, updates all information shown on displays at trailheads. Gets called by trailhead displays.
- *Locked Data: void*- Locks the day's data into the Black Box and cannot be deleted.

6. Development Plan and Timeline:



Start Date: March 21, 2023***

- March 23th-27th: Speak to client and set up a contract for 5 years minimum.
- March 27th- 30th: Evaluate Client's requirements
- April 1st-30th: Establish timeline, software architecture ideas and needed classes
- May 1st-31st: Establish attributes and operations as well as what server process to use.
- Month June: Head to park to gather information and set up cameras and displays. Confirm the class, attributes, and operations are correspondent to the client's needs.
- Month July: Construct Architecture diagram and UML diagram for better compatibility of build.
- Month August: program web pages using HTML and react. js and have base pages ready for the frontend. Connect ranger phones to server mainframe as well as display real-time connection.
- Month September: Make a database server using SQL for the backend server information storage. Create the login for rangers and staff.
- Month October: Create the classes and assign the classes to the pages and correct profile(ranger or patron).

- Month November: Assign web page pointers for string hyperlinks to be directed to different intended pages. Connect weather information to distinct trails and weather paths/ areas.
- Month December: Establish network security for camera and displays, and finalize network-server-Black Box connection. Create footage/sound/date allocation for server and Black Box storage.
- Month January: Finalize classes, attributes, operation uses, and directories.
- Month February: Finalize editing privileges, bug testing, server accuracy connection. Meet with clients to confirm requirements have been met.
- **Final Month: March 1st- March 21st, 2024**: Present software to the Tanzanian Government and Rangers of the park.

Estimated Completion Date: March, 21, 2024* (366 days)**

Base Test Cases:

1. Basic interactive system capabilities allowed for a single user Ranger/Staff and capable of reaching the system in real-time for moments notice information.
2. Directly able to update information through cell-phone with 70% accuracy success for 1 attempt, and 100% success within 4 attempts.
3. Lock Feature, makes all data read only till admin/ranger log in.

Team Responsibilities:

Matteo: SWA Diagram, SWA Description, Software Overview, System Overview.

Neel: UML Diagram, class descriptions, System Overview, operations descriptions.

Ryan: UML Diagram, attribute descriptions, operations descriptions titles, Timeline.