University of Nevada, Reno

AI-Driven User Experience and Accessibility Enhancements for a Sensor-Based Platform

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science
in Computer Science and Engineering

by

Nikhil N. Sharma

Dr. Sergiu M. Dascalu, Co-Advisor
Dr. Frederick C. Harris Jr., Co-Advisor

May, 2025

THE GRADUATE SCHOOL

We recommend that the thesis
prepared under our supervision by

Nikhil N. Sharma

entitled

AI-Driven User Experience and Accessibility Enhancements for a Sensor-Based Platform

be accepted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

Dr. Sergiu M. Dascalu,
Co-Advisor

Dr. Frederick C. Harris, Jr.,
Co-Advisor

Dr. Yantao Shen,
Graduate School Representative

Markus Kemmelmeier, Ph.D., Dean,
Graduate School

May, 2025

Abstract

The complexity and volume of IoT-generated data have grown rapidly, outpacing the ability of non-expert users to interpret and act on it to make real-world decisions. This thesis presents a prototype platform that integrates artificial intelligence to enhance both the user experience and accessibility in data-intensive environments. The system implements a conversational large language model capable of generating real-time, natural language summaries of various types of data visualizations-making complex information more accessible to users, particularly those relying on screen readers or keyboard-only interactions. A comprehensive assessment of the system and data interpretation capabilities demonstrates its reliability and preferred use by users.

Beyond data summarization, the portal enables further querying of data, allowing users to converse with AI using natural language to explore insights without needing expert knowledge on the domain. The interface possesses full keyboard accessibility design and a suite of guided help videos to accommodate a wide range of accessibility needs. Additionally, a map-based visualization layer was launched, allowing users to explore live sensor data and receive AI-driven insight related to the geographical positions of the sensor. The prototype has been applied to a platform for environmental monitoring, but demonstrates potential across domains such as agriculture, healthcare, financial management, and citizen science.

Dedication

I dedicate this thesis to my family.

Acknowledgments

I would like to thank my co-advisors, Dr. Frederick C. Harris, Jr., and Dr. Sergiu M. Dascalu, for introducing me to the graduate program and for their ongoing support throughout my graduate studies. I would like to thank Dr. Yantao Shen for providing additional feedback on this thesis. Additionally, I would like to thank my peers Chase Carthen, Kaden Nesch, Vinh Le, Zach Estreito, and Jehren Boehm for being a part of the team contributing to the project to which this thesis also belongs.

Furthermore, my thanks also go to the co-principal investigators of the NSF project that supported my work on this thesis: Dr. Scotty Strachan, Dr. William Savran, and Dr. Anne Heggli.

My heartfelt thanks go to my sister, parents, and friends, who have constantly supported my graduate studies. Without your backing, I would not have been able to make it this far!

This material is based upon work supported by the National Science Foundation under grant number OAC-2209806. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

Abstract i

Dedication ii

Acknowledgments iii

List of Tables vii

List of Figures viii

List of Listings ix

List of Acronyms x

1 Introduction 1

2 Background 4
- 2.1 VISTA Overview . . . 4
  - 2.1.1 Projects . . . 4
  - 2.1.2 Gateways . . . 5
  - 2.1.3 Organizations . . . 5
- 2.2 Limitations of the current VISTA portal . . . 7
- 2.3 Technologies Used in VISTA . . . 8
- 2.4 User Experience (UX) principles . . . 13
  - 2.4.1 User-Centered Design . . . 13
  - 2.4.2 Evaluation . . . 14

3 Related Works 17
- 3.1 IoT Data Representation . . . 17
  - 3.1.1 Common Techniques for IoT Data Visualization . . . 17
  - 3.1.2 Common Tools for IoT Data Visualization . . . 18
  - 3.1.3 Challenges in IoT Data Representation . . . 19
  - 3.1.4 Data Accessibility . . . 21
- 3.2 Artificial Intelligence for User Experience and Accessibility . . . 24
  - 3.2.1 User Experience Enhancements . . . 24

3.2.2 Accessibility Improvements and Optimizations 25
3.2.3 Challenges and Limitations 26

4 Motivation and Overview of Proposed Solution 28
4.1 Main Goals 28
4.1.1 Integrate Insight-Driven Tools for Better Data Understanding 28
4.1.2 Improve Accessibility for Diverse User Needs 29
4.1.3 Enhance User Experience Through AI Personalization 29
4.2 Intended Users 29
4.2.1 Researchers at the University of Nevada, Reno 30
4.2.2 Citizen Scientists 31
4.3 Approach Outline 32

5 Software Modeling and Implementation 34
5.1 Requirements Specification 34
5.1.1 Use Case Modeling 35
5.1.2 Requirements 36
5.2 Software Architecture 40
5.2.1 Flask Back End 42
5.2.2 The Things Stack 42
5.2.3 ChatCSV 42
5.2.4 AWS S3 44
5.2.5 Invoking the ChatCSV API 46
5.2.6 Suggestive Actions 47
5.2.7 Geographic Sensor Representation 51
5.2.8 Supplementary Features 52
5.3 Challenges 53
5.3.1 Prompt Engineering 53
5.3.2 Data Pipeline 55
5.3.3 Stochastic Responses 56

6 Application Scenarios 58
6.1 Interface Overview 58
6.2 Scenario A: Commercial Applications 66
6.2.1 Retail Intelligence 66
6.2.2 Financial Management 66
6.3 Scenario B: Essential Services 67
6.3.1 Smart Agriculture 67
6.3.2 Medicine and Healthcare 67
6.4 Scenario C: Civic and Educational 68
6.4.1 Neighborhood Weather Watch 68
6.4.2 Urban Gardening 68
6.4.3 Data Exploration for Students with Disabilities 68

7 User Study and VISTA+ Portal Evaluation **70**
7.1 Objective . . . 70
7.2 Experimental Setup . . . 71
7.2.1 Participants . . . 71
7.2.2 Setup . . . 71
7.2.3 Design . . . 72
7.2.4 Procedure and Questionnaires . . . 73
7.2.5 Tasks . . . 77
7.3 Results . . . 78
7.3.1 Tests Conducted . . . 78
7.3.2 Findings . . . 79
7.3.3 Participant Comments . . . 84
7.4 Discussion . . . 86

8 Conclusions and Future Work **89**
8.1 Summary of Contributions . . . 89
8.2 Future Work . . . 89

References **91**

List of Tables

3.1 Breakdown of some Web Content Accessibility Guidelines adapted with information from [24] 22

5.1 Use cases as derived from proposed user stories for the software. . . . 37
5.2 Requirements table for the proposed software. 38
5.3 Requirements traceability matrix mapping each use case to the requirements for software. 39
5.4 Example structure of the user events table in DynamoDB 48
5.5 Prompts and the associated challenges encountered in the response from the LLM. 54

7.1 ANOVA results for number of clicks by task level. 80
7.2 ANOVA results for number of clicks across all tasks. 80
7.3 ANOVA results for task completion time by task level. 81
7.4 ANOVA results for task completion time across all tasks. 81
7.5 Summary of two-way repeated measures ANOVA results with significance for task completion time, number of clicks, and accuracy. 82
7.6 Mann-Whitney U test results comparing execution time across proficiency levels. 83

List of Figures

1.1 Intergenerational inequity in terms of different climate hazards experienced by individuals born in 2020 compared to those born in 1965 [11]. 2

2.1 Data export page as seen on the VISTA portal. 5
2.2 Data visualization page as seen on the VISTA portal. 6
2.3 Device map page as seen on the VISTA portal. 7
2.4 High level diagram of the network architecture of the VISTA portal that shows the data flow from the LoRa end devices to the web interface. 12

5.1 New architecture displaying the changes for the VISTA+ portal. . . . 41
5.2 Diagram of processes for Suggestive Actions. 50

6.1 Homepage of the VISTA+ portal showing user suggestions. 59
6.2 Loading state on the data visualization page after user action click from the homepage. 59
6.3 AI Analysis feature on the export data page. 60
6.4 Chat interface where the AI response includes a graph of data. 61
6.5 AI Analysis feature on the data visualization page. 62
6.6 Chat interface where users can further query the model for a deeper analysis. 62
6.7 New map page on the VISTA+ portal with a sensor on the map. . . . 63
6.8 Chat interface on the map page with response related to geographic information. 64
6.9 Example help video buttons that trigger help video modals. 65
6.10 Help video modal on the add end device page. 65

7.1 Bar chart of computer proficiency ratings from pre-study questionnaire. 83

List of Listings

1 Example format of the ChatCSV API. 45

List of Acronyms

Abbreviation	Full Name
AI	Artificial Intelligence
ANOVA	Analysis of Variance
API	Application Programming Interface
AWS	Amazon Web Services
CSV	Comma Separated Values
EM	Environmental Monitoring
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDD	Intellectual and Development Disabilities
IoT	Internet of Things
JSON	JavaScript Object Notation
LLM	Large Language Model
LoRa	Long Range
LoRaWAN	Long Range Wide Area Network
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
NLP	Natural Language Processing
REST	Representational State Transfer
SDLC	Software Development Life Cycle
TTS	The Things Stack
UI	User Interface
UX	User Experience
VISTA	Visualizing IoT Sensor Tracking and Analytics
WAI	Web Accessibility Initiative
WAI-ARIA	Web Accessibility Initiative – Accessible Rich Internet Applications

Chapter 1

Introduction

Environmental monitoring (EM) is an essential step in mitigating environmental issues occurring from natural causes or human impact. With the growth in population and pollution levels, there is a strict rise in the need for natural resource management through EM. The Environmental Protection Agency's National Climate Assessment reports that a "person born in North America in 2020 will experience more climate hazards during their lifetime, on average, than a person born in 1965"[11]. Figure 1.1 reveals as many as 5 times more climate hazards than a person born in 1965. Aside from environmental concerns, EM as an industry has also experienced immense growth. The market size for global EM was valued at an astonishing 23.44 billion in 2024 and is expected to grow to over 25 billion in 2025 [19]. All these facts are indicators of the importance of research in the field of environmental monitoring and assessments.

Environmental monitoring involves data collection, analysis, and interpretation to understand ecological challenges fully. Data is often collected using deployed sensors made from Internet of Things (IoT) technologies. These devices record various types of data, including temperature, moisture, and humidity. The analysis can be performed by an interface that aggregates the data and allows data visualization in multiple formats. The interpretation is problem-dependent and usually requires that researchers be adept at the issue they are trying to resolve. Data interpretation is an issue that forms a barrier for individuals trying to solve real-world issues, and the use of Artificial Intelligence (AI) proves promising in solving this problem.

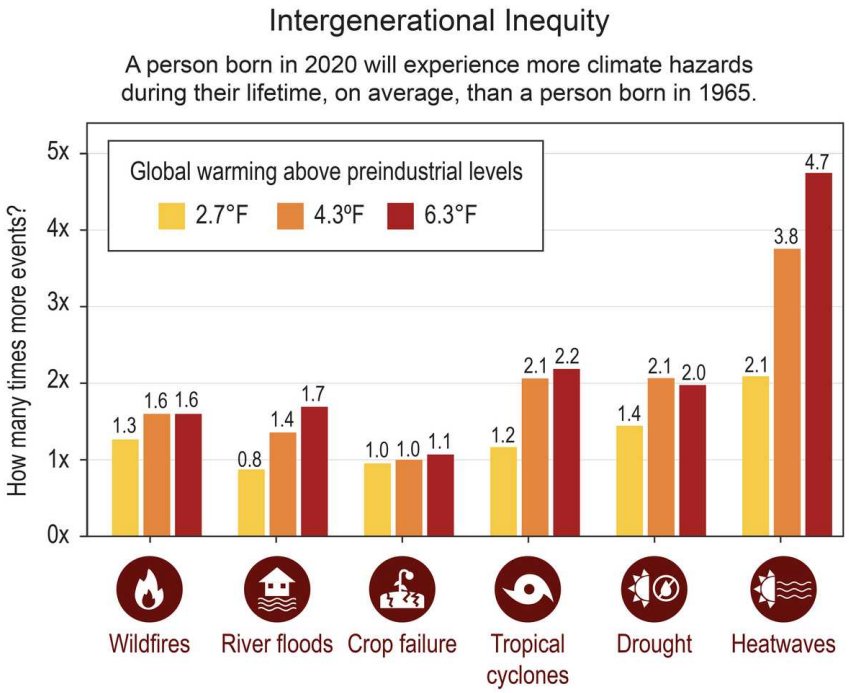

Figure 1.1: Intergenerational inequity in terms of different climate hazards experienced by individuals born in 2020 compared to those born in 1965 [11].

In recent years, AI has been a driving force behind significant technological advancements in "numerous applications across different domains, including healthcare, banking and finance, retail and product recommendations, education, energy, media and entertainment, and many other domains" [22]. The adoption of AI by everyday consumers and corporations has also seen steady growth (slightly over 18% on average [13] in 2024). The inner workings of AI: neural networks, deep learning, and machine learning have been a constant research and development for many decades, so Artificial Intelligence isn't a new concept. The development of large language models and ease of use have made it an incredible resource for anyone to access. With the use of natural language, consumers can ask various models for information on thousands of topics and get a response in a matter of seconds. So, how can AI be implemented to solve the issue of data interpretation in environmental monitoring?

This thesis aims to design and combine artificial intelligence with environmental monitoring by building features within the Visualizing IoT Sensor Tracking and

Analytics (VISTA) portal. This portal is already in development at the University of Nevada, Reno, part of the project "CSSI: Elements: Innovating for Edge-to-Edge Climate Services," supported by the NSF Cyberinfrastructure for Sustained Scientific Innovation (CSSI) under grant number OAC-2209806. For simplicity, the portal presented in this thesis will be referred to as the "VISTA portal". The VISTA portal integrates cutting-edge technology that allows for sensor deployment, management, and basic data visualization options. The VISTA+ portal extends this functionality to use AI for a wide variety of features that enable non-expert users to carry out environmental assessments. The research done in this thesis also aims to improve the accessibility and user experience to increase overall user satisfaction and retention.

The rest of this thesis is structured as follows: Chapter 2 provides background knowledge integral to understanding the material of this work. It includes an overview of the functions provided by the VISTA portal currently, the limitations, the current implementation technology, and a brief overview of the user experience principles that are referenced often in the thesis. Chapter 3 presents existing solutions for viewing IoT sensor data, the common techniques and tools used in data representation, and how AI has so far been applied to UX and accessibility. This chapter also briefly discusses data accessibility issues and how they can be remediated. Motivation and an overview of the proposed solution are given in Chapter 4. This is where the main goals, intended users, and the approach are introduced. Chapter 5 follows the Software Development Life Cycle to demonstrate the technical implementation of the system, discussing requirements, architecture, and the challenges during the development period. After describing system details, Chapter 6 speaks about the potential usage scenarios in multiple industries and conditions. Images of the final interface and features are provided. A user study was conducted as a part of this thesis, and the results are discussed in Chapter 7. Finally, overall research conclusions and future work are contained in Chapter 8.

Chapter 2

Background

2.1 VISTA Overview

The Visualizing IoT Sensor Tracking and Analytics (VISTA) portal allows users to view and manage their network of environmental sensors for environmental monitoring (EM). This section discusses its main features. The main modules of the VISTA portal are projects, gateways, and organizations. They serve as a way of accessing the features available to the user.

2.1.1 Projects

Projects are a way to organize a group of sensors. By accessing their project, users can manage sensors to add a new sensor, modify an existing one, or remove a sensor from their project. Projects allow the user to also view and analyze their data coming from their end devices. They can do this in two ways, either through the data export page (Figure 2.1) or the data visualization page (Figure 2.2). The data export page allows for sensor selection and viewing of data in tabular format. The table is feature-rich in terms of filtering, sorting, and overall organization. The user can also choose to export this same data in two different formats: CSV and JSON. The data visualization page also allows for sensor selection but visualizes the data in the format of a line graph where the x-axis represents time and the y-axis represents the applicable reading. Aggregation features for data visualization allow users to summarize data a little more efficiently. Users may also choose to set the location for their sensors, which

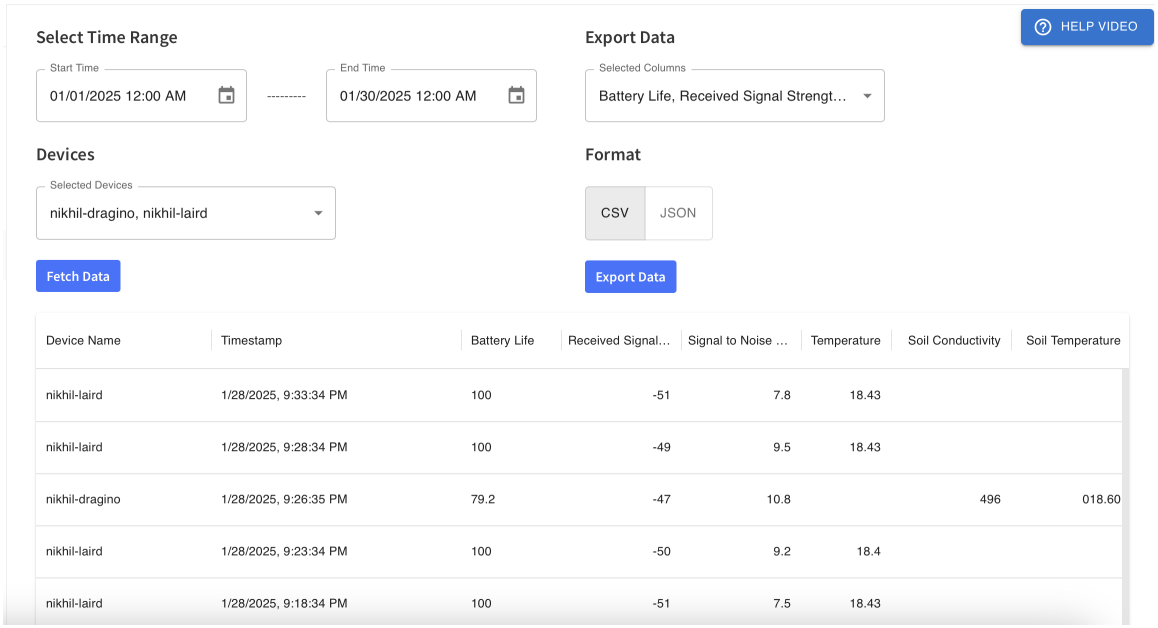

Figure 2.1: Data export page as seen on the VISTA portal.

they may do through the individual map pages after clicking on a sensor (Figure 2.3).

2.1.2 Gateways

The gateways module allows users to manage their gateways. A gateway in the sensor range is needed for the data stream to be complete. Gateways are the means of collecting data from multiple sensors and transmitting it to a unified database. After the connection between the sensor and the gateway is established, the gateway begins to send the data. Through this module in the VISTA portal, a gateway can be added, modified, or removed. The gateways module also allows the user to see the raw data being received from the physical gateway. Similarly to the projects module, allowing users to set sensor location, users may also set the location for their gateway.

2.1.3 Organizations

The organizations module provides a layer of abstraction for projects and gateways in terms of ownership and collaboration. Organizations can be added as collaborators to a project, therefore giving access to that project to all of its users. In the same way

Figure 2.2: Data visualization page as seen on the VISTA portal.

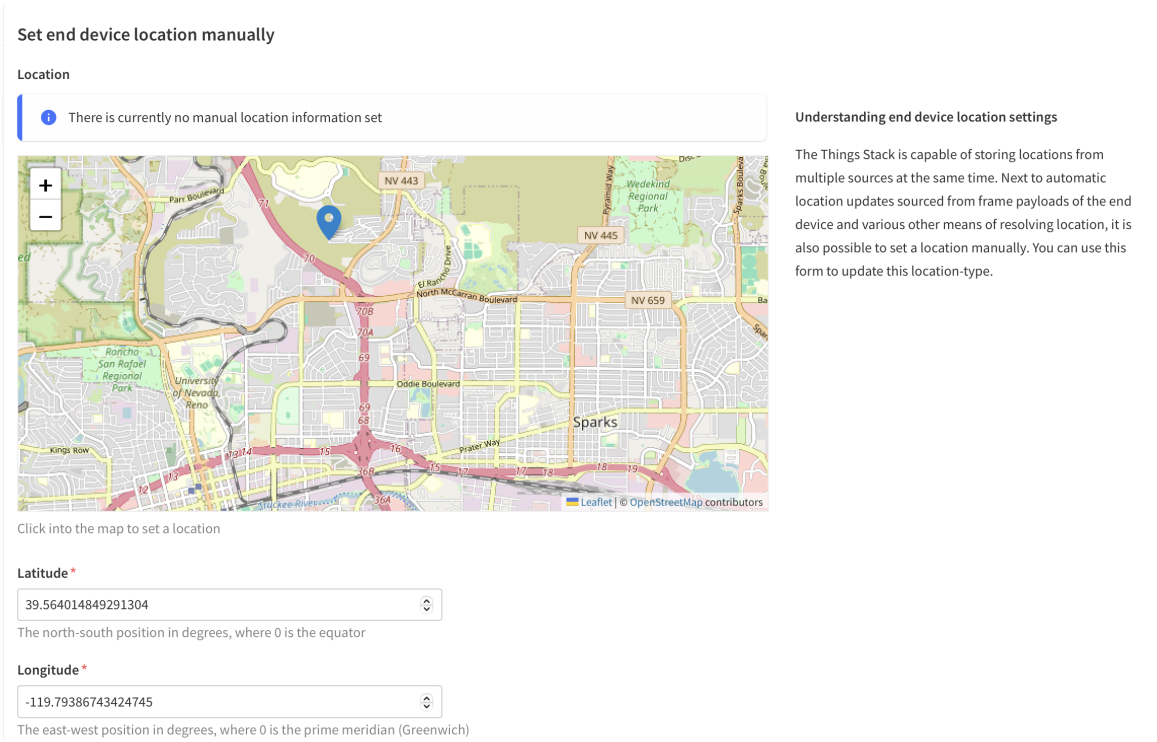

Figure 2.3: Device map page as seen on the VISTA portal.

that project access is manageable by organizations, gateways can also be assigned to an organization instead of a single user. The access can also be controlled using individual permissions so that partial access can be given to simply view a project or gateway without allowing modifications. VISTA portal admins have access to all projects, gateways, and organizations.

Along with the features outlined before, VISTA implements OAUTH as the first step to access the portal, followed by a simple login/registration process. Each new registration has to be approved by an admin to gain entry to the portal.

2.2 Limitations of the current VISTA portal

The VISTA portal is satisfactory in its features, providing users with methods to perform tasks related to environmental monitoring. Still, the portal has limitations in terms of accessibility, ease of use, and the lack of interpretation help for users.

Accessibility issues include a lack of website control using the keyboard, the ab-

sence of text alternatives for images, and insufficient color contrast. These are all important elements for people using screen readers to navigate the website with a logical flow. Full accessibility is not ensured with only the three considered improvements, but it is a step in the right direction in improving the overall availability of the portal for the disabled.

Difficulty in use could be associated with the interface being implemented without considering what interface elements users associate with ease of use. Good websites, or rather good applications, are made with the user in mind during every design process. The portal could be improved by redesigning certain pages with ease of use in mind for the user. This means adding shortcuts to popular features, redesigning pages with logical flow, and simplifying common tasks for faster use. These features and improvements are made for what the industry calls a "power user".

With the targeted users being researchers and citizen scientists, the portal currently does not guide for users to interpret their data in any way. Citizen science is the "intentional involvement, in a non-professional capacity, of people in the scientific process" [17], therefore, there needs to be some sort of process that'll allow this set of users to interpret the data without needing to substantially research the problem topic. This could be done in various ways, including adding help videos for confusing processes or adding tool tips for complicated terms.

2.3 Technologies Used in VISTA

The current Visualizing IoT Sensor Tracking and Analytics (VISTA) portal is implemented using LoRaWAN (Long Range Wide Area Network) as the primary method of data transmission. It is an attractive method for EM as it allows for long-range data transmission with little power consumption. The interface and overall connectivity are managed using a modified version of the Things Stack (TTS) application made by the developers at The Things Industries. The Things Stack Open Source Edition is an open source project under the APACHE 2.0 license that allows for modifications to the application. The base functionality for The Things Stack does not include ways

to view the data, which means that the data export and data visualization pages were added by the team of developers working under the grant mentioned before. This modified version of The Things Stack is what is referred to as the VISTA portal. The VISTA portal comprises two back-end applications and a front-end application for its functionality, which is detailed hereinafter.

- **Back-end provided by The Things Stack:** The base version of the back-end provided by The Things Stack is coded in the Go programming language and is quite powerful. It features a robust API that allows developers to tie into the management of projects, gateways, and organizations without touching a user interface. The features provided by this back-end support all the functionalities provided by the base version of the front-end as well. Additional features include integrations with other technologies such as AWS IoT, Azure IoT Hub, and Google Cloud IoT Core.

- **Back-end created by developers on the grant:** The implementation of a separate back-end using Python in the form of a Flask app was decided to keep the tech stack fairly simple. This back-end's main purpose is to allow for ways to view data to the user in the front-end, a key feature not possible with the base implementation of The Things Stack. A range of APIs was built. Some examples include those that communicate the type of sensors being added to the ThingsStack and others that return data formatted to plug right into the data export and data visualization pages in the front-end. To understand why this back-end was necessary, one must understand the complete data flow.

  A complete data flow diagram is depicted in Figure 2.4. Much of this information is derived from a paper written by the developers on the grant here [7]. Here are the steps taken from the point at which the data is generated to the point at which the data can be seen by the end user.

  1. **Data is generated by the sensors (LoRa end devices)**. These sensors can be of different types (soil moisture, temperature, distance, humidity,

etc.), so they will have different ways of measuring the value. For example, a temperature sensor might have a probe attached to the sensor, but a distance sensor might have sonic sensors. The sensor data also varies in the default transmission frequency, but oftentimes this can be altered.

2. **Transportation to gateways.** After the value is recorded by the sensors, the data is transported to a gateway. As mentioned before, The Things Stack (TTS) utilizes the LoRaWAN network for data transmission. The LoRaWAN network allows for low-power, long-range data communication to the gateway. A nicer gateway will allow for 1-2 miles of range in urban areas and as much as 6 miles in rural areas. The gateway is connected to the internet using either a wired or wireless connection.

3. **Transportation to TTS and final destination.** The gateway sends data to The Things Stack server via the User Datagram Protocol (UDP) over the internet. TTS is built to merely act as a middleware for the final destination of the data. It is built so it can specifically receive LoRaWAN streams and transmit to the final destination as needed by the project or use case. The Things Stack is meant to be set up by a team of developers for a specific use case and deployment. This team will be referred to as the System Admin, henceforth. The system admin can choose to set the final destination through a message queuing telemetry transport (MQTT) server or one of the integrations built by The Things Industry (the team that manages The Things Stack). For this project, the decision to implement an MQTT server was made to allow for custom and better control over the data stream. The MQTT server transports data to a custom Timescale database, and this is the permanent location at which the data will be available. Timescale is built on PostgreSQL but is meant to handle time stream data more efficiently.

4. **Viewing data through the portal.** Since the final destination is differ-

ent for every case of TTS, there can't be a uniform data viewing solution on the platform by default, as each instance of the TTS for each use case will be completely different. So, to accommodate the need to view data on the platform, the team of developers created a separate backend to query data from Timescale and be able to view that in the front-end. The user navigates to one of the pages that they may view data on (see Figure 2.1 and Figure 2.2) and then proceeds to follow the steps on the page to select a time frame along with other options. When the user presses submit, the front-end sends an HTTP GET request to the appropriate Flask back-end API, which routes to a function that queries data and returns it to the front-end. The data returned to the front-end then plugs into the elements present on the page to give the user either a beautiful graph or a detailed tabular form of data display.

- **Modified front-end using The Things Stack:** The front-end is implemented with the React JavaScript framework and transpiled via webpack using babel to be interpreted by the browser.

  The data export page implements the tabular form of data by utilizing the Data Grid offered by Material UI. With careful implementation of this data grid, the columns can be sorted, filtered, and searched. There is little to no lag, even with a large number of rows for this data grid.

  The data visualization page was built with a combination of native and Material UI elements. The graph is a line chart built using a relatively recent and experimental version of x-charts offered by MaterialUI. While this accomplishes the basic needs to visualize the data, user experience is being sacrificed as the graph tends to lag with over 2000 points being displayed on the graph. The cause has been determined to be the SVG rendering for the graph, but it is something done by the library and out of the team's hands for now.

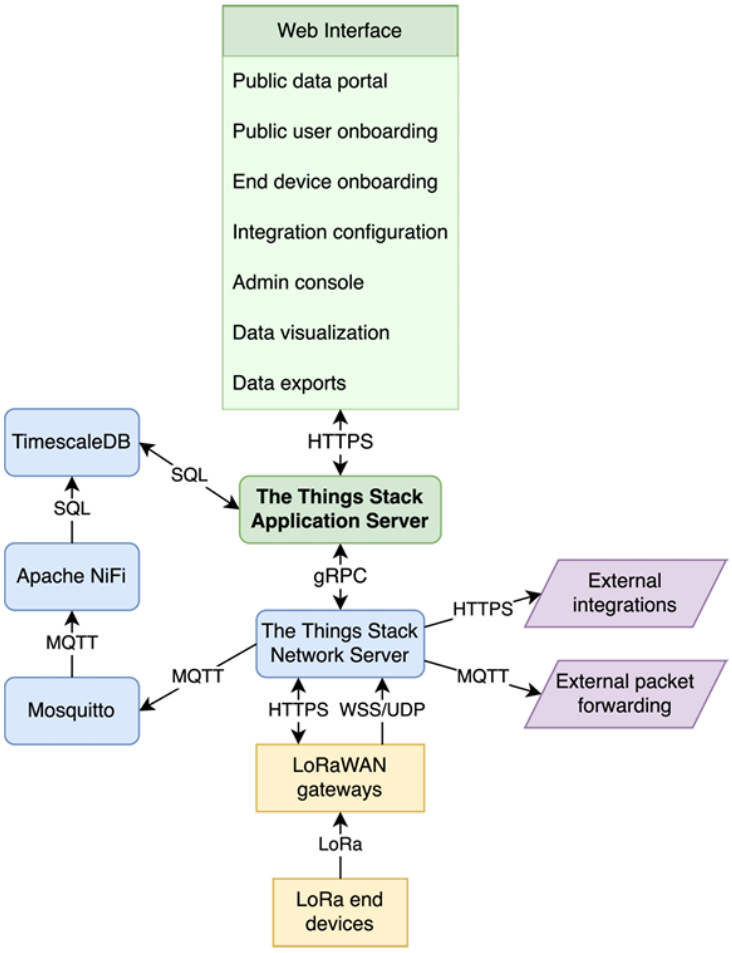

Figure 2.4: High level diagram of the network architecture of the VISTA portal that shows the data flow from the LoRa end devices to the web interface.

2.4 User Experience (UX) principles

User experience (UX) is a diverse field that integrates many different research areas such as human-computer interaction (HCI), psychology, product design, and development. Though hard to truly define, UX goes "beyond normal usability and functionality aspects of products by incorporating the users' feelings and emotions towards these products before or during interaction" [3]. This thesis aims to improve the UX of the VISTA portal, and so it is important to understand what that means.

2.4.1 User-Centered Design

A product may be considered usable early on when minimum functionality has been achieved, but how can developers take that to the hedonic level and get their users to maximum adoption? [20] states that a positive user experience (therefore higher adoption) can be achieved if a user-centered design process is followed during development. A user-centered design is a software design methodology for developers and designers that follows a simple goal: to have the application meet the needs of its users. User-centered design comprises four activities as described in the ISO 13407 standard and found in [20]:

1. To understand and specify the context of use.
2. To specify the user and organizational requirements
3. To produce design solutions.
4. To evaluate designs against requirements.

To improve the UX of the VISTA portal, these four activities will be elaborated in chapter 5. The systematic review of user-centered design done in [20] reports how a graphical user interface (GUI) may be redesigned to follow a user-centered design. The top result was prototyping. Prototyping is a "technique that aims to design the initial or 'draft' version of a system, which allows the designers to explore their

ideas and show them to end users before investing resources into development". This means that prototyping a product may provide good feedback from stakeholders and allow for another iteration of design/implementation to improve product adoption. Prototyping can be done via the use of popular UI design programs such as Figma or by simply implementing the UI and having users give feedback based on that.

Outside of development, the systematic review in [20] also reports that the identification of stakeholders/end users may be beneficial in reaching a good user experience for a product. Other ways may be usability evaluations of the original GUI and interviewing potential users to identify the needs of the users. All these techniques can be used together for another iteration of development/design for the VISTA portal to achieve a better overall user experience.

2.4.2 Evaluation

Evaluation is measuring how well the product is doing in terms of user experience. For designers and developers, it's a key concept that allows them to make further improvements to any project in software engineering. So, how does one evaluate user experience?

If a user is performing a task faster in a redesigned experience compared to the original experience, it can be assumed that the redesigned experience has a more positive UX. Other factors that seem obvious in evaluating for a better UX are less number of clicks, and higher overall accuracy for the task. Though [3] mentions that "objective measures such as task execution time and the number of clicks or errors are not valid measures for UX." Ultimately, how the user feels about the system must also be measured along with the objective measures. There are several challenges in evaluating user experience. Some of them include the dynamic nature of different user experiences, the unit of analysis, and the fragmentation and complexity [3].

- Dynamic Concepts: Depending on the experience that is being evaluated, a different range of concepts might be used. For example, in the case of evaluating the experience of a shopping application that wants the end user to purchase

a product, the evaluators might focus on the aesthetic variables that make the user want to keep using the application to shop. In the previous example, the evaluation may change once more depending on the product that is being sold. Other variables include emotional, affective, experiential, and hedonic. These variables are omitted or included in the evaluation depending on the context.

- Unit of analysis: This is another challenge in the evaluation of user experience because not all user experiences are defined by the same sort of interaction. UX can be analyzed from the perspective of an individual end-user interacting with a standalone application or the perspective of multiple end-users interacting with multiple applications. This creates a challenge in the way that the evaluation must then be modified to accommodate the different modes of interaction, with the possibility of introducing more complex methods of evaluation

- Fragmentation and Complexity: [3] states that the "scene of UX research is fragmented and complex by dissimilar theoretical models with diverse foci such as pragmatism, beauty, affect, experience, value, pleasure, emotion, hedonic quality, etc". This hints that UX evaluation is a complex subject with many different models that can be applied depending on the context.

Another great challenge in evaluating user experience is time. Time is an important factor in conducting experiments to evaluate user experience, as it can actively change results. The four time spans in which UX may be evaluated are before usage, during usage, after usage, and over time [3].

- Before Usage: Before usage is anticipated, UX or otherwise, what the user expects the experience to be like. Anticipation can be formed from related technologies, brand, advertisements, and other people's opinions. Research demonstrates that anticipated UX can have an extended impact on subsequent experiences, not necessarily positive or negative [3].

- During Usage: During usage is when the evaluation occurs as the user is actively using the final application. This is an important phase in evaluation, and researchers emphasize the need to evaluate the UX while users are interacting with the final product, as it allows the researchers to measure the change in emotion.

- After Usage: This is simply the user reflecting on the experience at a time after the usage has occurred.

- Overtime Usage: Overtime usage gives evaluators a cumulative evaluation of an application. Users will tend to change their opinion after interacting with other similar applications or interacting with the subject application multiple times, and so this is a great way of evaluating the overall user experience.

Overall, the evaluation should be tailored to the context of the user experience and the goals of the evaluator.

Chapter 3

Related Works

3.1 IoT Data Representation

The current VISTA portal includes two ways of viewing data, a tabular data grid and a line-graph-based chart. Internet of Things devices create large amounts of data, and it is important to see what tools and techniques are being used to represent this data. This will help analyze the current solution and software engineer changes that will help achieve a higher user experience. This section summarizes current industry or project-based IoT data visualization techniques, tools, and challenges.

3.1.1 Common Techniques for IoT Data Visualization

Data visualization is any type of graphic that can present data to the end user in a visual way that will help the user understand the data better. One of the most common ways for data visualization is the use of charts. Charts are used depending on the context and the data that is being visualized. Here are some of the charts characterized in [18].

- Line Charts: These are continuous curves that may be used to depict dependencies between output variables. Common uses include the need to view historical or time-series data. An example is the plotting of time and temperature, as they both depend on each other for meaning.

- Scatter Plots: These visualize data points in two-dimensional or three-dimensional

space. Scatter plots are common in machine learning, where large amounts of data go through principal component analysis so it can be represented while still maintaining the important patterns and relationships.

- Bar Charts: Bar charts or bar plots utilize rectangles to show the relationship between numerical (frequency most time) and categorical values. Bar charts are a "fine-grained approach to reveal the parameter distribution over time, while the range of values is set into a series of intervals."

Another notable method of data visualization that isn't strictly a chart is maps. Maps can be used to represent IoT data that is solely geographical or contains geographical elements. An example of this is IoT-based sensors recording environmental variables while deployed in the outdoors and represented on a map. A user viewing the map can look at their data and sensors to make reasonable assumptions as to why the data being recorded is so different. An extension of the maps presented in [18] is heat maps. Heat maps are another great way to view the spatial distribution of a variable, and they also provide a more aesthetically appealing perspective.

3.1.2 Common Tools for IoT Data Visualization

Many different tools exist to generate the charts and visual methods discussed in 3.1.1. The tools differ in their respective abilities, cost to use, and integration capacity. Here are some of those tools and their different characteristics taken from [15, 18].

- **Tableau**: Tableau is a common name heard in data visualization and is known for its fast and flexible data visualization tool. With the input of data, a user can generate multiple different types of graphs and charts. One of Tableau's key features includes allowing R scripting, which allows for very complex data analysis. The simplest form of data connectivity is using an Excel file, while some of the more complex data integrations include Amazon Aurora, Cloudera Hadoop, and Salesforce. Tableau is known to handle large amounts of data while still being an effective tool for data analytics. Tableau is not free for

users, but it offers limited usage with a small amount of free online storage available for its users.

- **Common CSV Programs**: Programs for comma-separated value (CSV) files, such as Google Sheets or Microsoft Excel, are very capable of generating visuals for data. They offer the ability to select data right from their platform and generate different kinds of charts from that. Though these charts are often quite basic and require modifications to generate useful visuals. They also offer integrations of data from popular data storage services such as SQL servers, cloud servers, and publicly available data sets. The greatest advantage to using common CSV programs for data visualization is that these programs are usually free to use, with a very small number of features hidden behind a paywall.

- **Grafana**: Grafana is an advanced data visualization tool that focuses on time-series data and monitoring over a period of time. Grafana's data integrations are plenty, with the most notable being real-time data connectivity. This tool basically acts as a dashboard for users to view real-time data visually. Users can also set up alerts if the real-time data has an anomaly. It has many different charting capabilities and has many techniques for advanced data analytics. Grafana is open-source (free to use), but the platform itself takes some time to get familiar with.

3.1.3 Challenges in IoT Data Representation

IoT data representation is an important part of the IoT data pipeline, and so it is important to consider the challenges that are presented while trying to visually exhibit IoT data. Visual analytics is a significant field that produces actionable insight, yet, "most visualization tools exhibit poor performance in terms of functionality, scalability, interaction, infrastructure, insight creation, and evaluation" [18].

Functionality challenges are caused due to the sheer volume of data sets, their high dimension, and their streaming nature [18]. Here are some functionality chal-

lenges [18]:

1. Diverse data sets are a challenge, as plotting that data requires assumptions that are not possible without full comprehension of the context.

2. Data sets being too closely related cause visual noise that makes it harder to interpret the visualization. Users are then unable to recognize patterns, hence making the visualization useless.

3. Presenting large data sets is a challenge since managing the aspect ratio, device resolution, and perception limitations of the user are all very tough problems to solve.

4. IoT data is often generated in real-time, so having infrastructure capable of supporting real-time data transmission is also a challenge.

5. Oversimplification of data is an unapparent challenge as well. Oversimplification misleads the viewer of the IoT data representation, often causing misinterpretation and a lack of complete understanding of their data.

6. Another overlooked challenge, yet one of the most important ones, is data overloading or over-plotting. This causes platforms to lag and become unusable. Proper data cleaning and dimensionality reduction could be used to overcome this specific challenge.

Scalability issues arise from system design errors when constructing the IoT data pipeline. Oftentimes, systems are designed to initially test that pipeline with a small volume of data without any consideration of what that might scale to in the future. A data visualization technique (line chart, for example) may work for a small volume of data, but it is important to consider if that same technique will be effective for larger amounts of data.

Interactivity challenges are related to having interactive elements available for the user as a part of the visualization. For example, for a generated line chart, users

should be able to interact with that chart by being able to modify the axes, zoom in to view granular data, or hover over parts of it to view the individual values. [18] mentions that the literature review shows that interactivity as a part of data representation is still challenging.

Infrastructure has been a common challenge mentioned in bits previously, but it is significant to mention that it is the core of powerful visualization systems. With the diversity of data being transmitted through IoT devices, it is hard to maintain a system infrastructure that can support the wide range of data. At the bare minimum, an application should have an infrastructure capable of supporting varying amounts of data with moderate support for types of data.

Insight creation goes back to the main purpose of having visual data representation: to help the user interpret the data. This goal is accomplished only if a meaningful visualization has been created and that it results in maximum "human capacity, and perception, to facilitate data exploration, [and] analytical reasoning"[18].

The final challenge mentioned in [18] is evaluation. As hinted in Section 2.4.2, evaluation is very challenging because of the dynamic nature of user experiences. However, to create better visualization systems, there must be a moderate evaluation of the GUI/interface for its effectiveness and trustworthiness.

3.1.4 Data Accessibility

Data accessibility is a prominent issue that can impact nearly 15% of the world's population - those who have some sort of disability [25]. Specifically for students with Intellectual and Developmental Disabilities (IDD), data visualization may not translate to a reasonable interpretation compared with people who do not suffer from those symptoms. While most accessibility research focuses on visual impairment (which is also important), there are still many unknowns for data accessibility or visualization accessibility for people with IDD. This subsection will go over (1) general accessibility guidelines provided by the World Wide Web Consortium (W3C) and the Web Accessibility Initiative (WAI) and (2) the guidelines provided in [25] to design

data visualizations that may better support individuals with IDD.

The World Wide Web Consortium (W3C) is an international organization that develops standards and guidelines for online presence with an emphasis on accessibility guidelines with the Web Accessibility Initiative (WAI). They have many resources and documents to explain how to make web content more accessible to people with disabilities. Here are a few of those guidelines hand-picked to relate to data visualization and IoT web portals, along with examples of how those guidelines could be implemented:

Table 3.1: Breakdown of some Web Content Accessibility Guidelines adapted with information from [24]

Guideline	Explanation	Example
1.1.1	All non-text content that is presented to the user has a text alternative that serves the equivalent purpose.	For graphs or charts generated on the portal, there exists a text alternative that summarizes the chart. Even better if the same text can highlight notable patterns and relationships.
1.3.1	Information, structure, and relationships conveyed through presentation can be programmatically determined or are available in text.	Similar to 1.1.1, where any information presented is also available in text if it is not already text-based.
1.3.2	When the sequence in which content is presented affects its meaning, a correct reading sequence can be programmatically determined.	This guideline has to do with the flow of forms and text on the page. Making sure that content is placed on the page in order that makes sense.
1.3.4	Content does not restrict its view and operation to a single display orientation, such as portrait or landscape, unless a specific display orientation is essential.	Making sure the content is in the correct order and functionality, regardless of screen size or orientation.
2.1.1	All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes	Making sure all forms, text content, and buttons are accessible using only keystrokes from the keyboard.

The rules provided by the W3C are greatly useful in making sure people with

disabilities have fair access to general online content. Wu et al. [25] took this one step further by examining guidelines suggested by other papers to form preliminary insight into accessible visualization for people with disabilities, specifically Intellectual and Development Disabilities (IDD). They studied chart types and embellishments as part of the attempt in the design process to achieve accessible visualizations.

- **Chart Types**: In the paper, they discuss that they tested various chart types for both time series data and proportional data. Their results concluded that people with IDD and without both preferred familiar representations for the most part. The notable takeaway for charts is to avoid using pie charts as they cause struggle among the IDD population in estimating correct quantities.

- **Embellishments**: Embellishments were further divided into abstract marks, chart junk, and icons. Abstract marks are when charts, for example, use simple shapes to convey data, showing rectangles on a bar graph instead of something fancier that would be assumed to be more engaging. Chart junk consisted of a single hue background image for the purpose of their experiments. Icons are a part of the markings on a graph. An example of icons could be using dollar signs, where there would be small dots on a line chart. Qualitative results indicated that embellishments may "create higher engagement and interest" [25], but overall, the results seemed to be mixed for the use of embellishments in accessible visualizations.

The paper found out that "while designs could significantly improve performance for people with IDD", "designs improving accessibility did not degrade performance for the control population" [25]. This suggests that designers and developers can continue to try to improve accessibility in visualizations while not sacrificing the universal interpretation of the graphics. These solutions and suggestions aren't set in stone, but they at the very least provide a preliminary synopsis of the realm of data accessibility and accessible visualization techniques.

3.2 Artificial Intelligence for User Experience and Accessibility

Artificial Intelligence (AI) has been a catalyst for technological advancements and is completely disrupting the ideas of traditional human-computer interaction. With the recent growth in everyday use of AI, it can only be assumed that AI can also be adapted for optimizing user experience and accessibility. This next section will examine the current scene of using AI for enhancing user experience and accessibility, followed by some challenges presented by the same.

3.2.1 User Experience Enhancements

The employment of AI for user experience enhancements is not something novel. Intelligent interactive systems, including "virtual assistants, recommender systems, intelligent help systems, and intelligent tutoring systems" have all been developed as an ongoing effort of applying AI for UX needs [22].

- Recommender Systems: Recommender systems are commonly used in the industries of entertainment, shopping, and finance. These systems are a "type of information filtering system that aims to predict the preferences or interests of a user, and recommend items or products that the user may be interested in" [22]. While referenced as AI, under the hood, recommender systems are machine learning (ML) algorithms that are fed products and items that the user has already seen or interacted with. The ML algorithm then returns similar products by using previously learned patterns from other consumers and data. In the case of recommender systems, AI is working to improve the user experience by alleviating cognitive load caused by the hassle of trying to manually find what the user is looking for.

- Virtual Assistants and Help Systems: Advanced virtual assistants or 'chatbots' have been the result of recent AI developments and have been spreading throughout industry, becoming the norm for communication between end users

and online services. Equipped with natural language capabilities, virtual assistants allow end users to talk to them as if they were talking to a human. They help reduce the cost to companies by having to hire fewer customer support staff. The latest virtual assistants like Google Assistant and Amazon Alexa can allow consumers to perform various tasks such as scheduling meetings, making calls, and interacting with smart devices, all without needing to lift a finger.

- Intelligent Tutoring Systems (ITS): These are systems that are used to teach a certain domain to its users. AI has improved this system by personalizing and adapting learning experiences for individual users. Algorithms analyze individual "user data and behavior to identify learning strengths and weaknesses and to provide personalized instruction and feedback" [22]. These systems can therefore provide customized learning programs, assessments, and exercises that will optimize the learning progress for each learner. Connectivity with Large Language Models (LLMs) can also help the user of ITS systems ask specific questions and get additional help as needed.

In the specific realm of data interpretation and summarization using AI, not many resources exist, as this topic is in a preliminary phase.

3.2.2 Accessibility Improvements and Optimizations

Accessibility improvements using AI are a relatively new field that presents great opportunities for the inclusion of people with disabilities related to accessing online content and services. The state of web accessibility today revolves around AI-based image recognition, text processing, and voice-based commands.

AI image detection can greatly empower disabled individuals by providing text descriptions from images that people would otherwise not be able to interpret. An example of this is "Microsoft's CaptionBot [which] compares uploaded images with millions of related images indexed by the Bing search engine, to provide automatically generated descriptions of the images"[1]. This kind of technology enables people with

screen readers to understand content from images without text alternatives.

Text processing using AI is another tool that exists today, which allows users to perform commands like text summarizations, simplification, and text splitting (splitting long sections into readable chunks). Text adaptations solutions enable people with cognitive disabilities to help their own learning and reading [1].

AI-based voice commands are another revolutionary development in the world of accessibility enhancements using AI. Users with vision impairments can speak certain commands into a microphone for certain tasks to be performed. Machine learning models capable of voice recognition translate the voice command into an actionable task, such as navigating pages, clicking buttons, or modifying screen options [1].

3.2.3 Challenges and Limitations

The topic of using artificial intelligence for improvements in UX or accessibility introduces new challenges and limitations. One great challenge is simply the guarantee of accuracy when using AI tools. Oftentimes, algorithms with AI make assumptions for the user. AI generates "hypotheses about both the user's present state of mind and their desired state, which may differ" [22]. This can lead to misinterpretations or misunderstandings for the user, causing the user more harm than benefit from using the AI tool. Privacy is another challenge when using AI, as it is susceptible to cybersecurity attacks that may put the user's personal data at risk. Finally, ethical issues also arise when content returned from AI models presents discriminatory or ethical issues. This could be reciprocated by mentioning that users could also use AI content for misuse [22].

Without a level of consistent accuracy from artificial intelligence, consumers are unlikely to trust AI systems. Loss of confidence in AI technology can severely hurt the overall sentiment from users towards a product, therefore negatively impacting the user experience and defeating the purpose of implementing an AI-driven enhancement. Regardless of just accuracy, AI models are biased given that they are trained on real-world data, which can sometimes contain biased content [22]. The potential for bias

must also be considered when designing systems using AI, as that will also build distrust in the user's eyes.

Popular Large language models (LLMs) are generally vulnerable to information extraction attacks, and oftentimes, companies will use centralized data storage solutions that are susceptible to data breaches. This can cause individual users to become victims of cybersecurity attacks by having their sensitive information stolen. Another related concern is the lack of transparency about how AI tool providers are using user data. This could mean that whatever a user decides to share with an LLM could potentially be shared with a third party. System designers and programmers need to be cautious of this challenge as they design software solutions that employ AI for the betterment of the user experience within their product.

Ethical issues must also be considered as a challenge while designing AI systems for UX or accessibility enhancements. As a reminder, AI can inherit biases from training data if it is incorrectly filtered, and therefore lead to wrongful decisions by the users of AI. An example of this is the use of AI in job-seeking platforms. Employers tend to filter applicants using AI as that saves them the time of having to manually review resumes, and it also speeds up the hiring pipeline. Though if the AI that they were using were potentially biased against a certain quality in applicants, that would make the whole process unfair for a certain group of people. On the other hand, consumers can also misuse AI-generated content for unethical purposes like identity theft or intellectual property theft.

Limitations exist in the form of costs to implement AI solutions for UX and accessibility enhancements, and for the abilities offered by AI models today. Training and continuously researching advancements in AI models, such as LLMs, is expensive, and therefore, that cost is distributed and managed by charging customers to use APIs to interact with the model or to limit the usage per consumer. AI models can perform better than ever and continue to improve day by day, yet they have their limitations in overall performance. Both cost and AI model abilities are important considerations before building software solutions for UX and accessibility enhancements.

Chapter 4

Motivation and Overview of Proposed Solution

This chapter introduces the proposed solution, intended users, and approach outline after researching the related works in Chapter 3.

4.1 Main Goals

The VISTA application is meant to be utilized by anyone wanting to implement LoRaWAN technology to solve a problem. The research and implementation done as a part of this thesis transform the VISTA portal from an exploratory application to one that allows for a personalized user experience full of AI-driven features that grant data accessibility and robust data analysis. This new proposed and modified version of the VISTA application will now be referred to as VISTA+.

4.1.1 Integrate Insight-Driven Tools for Better Data Understanding

By coupling modern large language models (LLMs) and existing aggregated sensor data, the application brings insightful data interpretation capabilities to its users. With greater data insight within the application, the portal reduces the need for self-data processing by researchers, therefore increasing user retention. The data-focused natural language responses introduce a paradigm shift in user experience and human-computer interaction, transforming static visual analytics into a conver-

sational environment that leaves users with an elevated sense of engagement.

4.1.2 Improve Accessibility for Diverse User Needs

Meeting modern accessibility standards (see Table 3.1) is the bare minimum any online tool should strive to meet for making an inclusive application. Another goal of this thesis is to adhere to or make progress towards established accessibility rules and leverage the use of AI to go beyond compliance in the effort to make the VISTA+ portal usable and inclusive to all individuals. Some Web Content Accessibility Guidelines (WCAG) have been implemented, while some have been greatly improved within the application. This thesis further develops the work mentioned in Subsection 3.2.2 to provide AI-driven text-based visual summaries, allowing present-time screen-readers a method of explaining the visual graphics on the page to those with disabilities.

4.1.3 Enhance User Experience Through AI Personalization

This final goal pertains to making the portal intuitive with a user-centered design focus for new features. The approach to this goal stems from the recommender systems studied during the research phase of this thesis in Subsection: 3.2.1. User activity could be monitored and then fed into a machine learning algorithm that could predict what the user might want to do next based on factors such as last activity, time of day, and interaction metrics. If implemented correctly, it could ultimately lighten the cognitive load from the user's perspective, removing the hassle of manual navigation and mundane tasks that users must regularly perform on the portal.

4.2 Intended Users

Intended users of the VISTA+ portal include researchers, citizen scientists, and virtually anyone intending to solve problems using LoRaWAN technology. The portal may be used by anyone, but a basic knowledge of IoT devices and access to LoRaWAN devices may be necessary to get started. The portal could also provide value to large entities such as the Department of Public Works to oversee projects that generate

large volumes of LoRaWAN sensor data. Here is a breakdown of two of the common intended users of the VISTA+ portal:

4.2.1 Researchers at the University of Nevada, Reno

The first proposed role is that of a graduate-level researcher at higher-education institutions such as the University of Nevada, Reno. This role brings background knowledge about how environmental assessments function and what the data can do for them. At the very least, the graduate researcher has read several pieces of writing that enable them to perform their research and analysis given a data set. They are relatively informed on environmental issues that occur daily and how those might impact readings from various types of sensors. The current VISTA portal solution allows them to deploy their sensors and see data flowing into a data visualization solution. Advanced researchers may obtain the raw data from sensors and use programming languages and libraries (Python and Pandas, for example) to process the data to make real-time decisions. Intermediate-level researchers might take the raw data and use platforms such as Tableau, Microsoft Excel, or Grafana to visualize the changes in a manner that makes sense for the topic they are studying. The changes as a part of this thesis could help replace the need for external programs, such as the ones aforementioned, by bringing the tools necessary for the user to analyze their data directly to the data export mechanism on the VISTA+ portal. The full extent of this role is better demonstrated through an example:

A scientist deploys several soil moisture sensors on a mountain to study water on snow levels throughout the year. With the help of the portal, the scientist can set up the sensors and begin to view data. From here, the researcher could take the raw data and try to analyze it via (1) traditional data analysis or (2) use one of the data visualization tools on the portal. For (1), they would begin by exporting the data and importing it onto a time series graph showing the relationship between time and moisture in the ground. This would then allow them to survey how snow melt infiltrates the soil over time, identifying anomalies related to climate variability, and

detecting general seasonal patterns. For (2), the built-in visualization tools on the VISTA portal offer limited capabilities compared to the traditional approach, but are more time efficient as they don't involve any cleaning or processing to view graphical displays of data. With the innovations brought by the thesis, the researchers studying water on snow levels are excited to utilize the AI analysis built on the portal, as that permits them to focus their time on the scientific process rather than the technical efforts of manual data gathering and viewing. They would be able to ask specific questions such as "What are the major trends in soil moisture over March?" or to explain relationships between multiple sensors such as "Is there a correlation between water on snow levels for the sensor placed on top of the mountain and the sensor placed at the basin?" These are only some examples of questions that are possible with the extent of artificial intelligence today.

4.2.2 Citizen Scientists

The second major role of the intended users of the VISTA+ portal is citizen scientists. These are people who are intentionally involved in the scientific process, usually for their own needs. Contrary to the other role, citizen scientists don't necessarily possess the background knowledge needed to fully carry out an environmental assessment. They can view data but may have difficulties interpreting the data in a meaningful manner. Here is an example of citizen science in action:

Bob, the homeowner, plants vegetables all around his backyard, hoping to have fresh vegetables in a couple of months. He has a busy schedule, but manages to water his new plants regularly for an equal amount of time for all plants. Several weeks pass, and he notices that some of his plants aren't doing as well as others, and starts the journey to figure out why his plants aren't growing at the same rate despite watering them equally. He finds the VISTA+ portal and deploys a couple of soil moisture sensors in parts of his backyard that are doing well for plants and the parts that aren't. After letting the data aggregate for several more weeks, Bob visualizes his sensor data on the portal and realizes that soil moisture values have

been severely low for the plants that have not been growing rapidly. At this point, Bob has no clue what might be causing this phenomenon to occur. This halts his citizen science process because he is not a professional in the field of environmental monitoring. With the natural language artificial intelligence updates to the portal, he is able to converse with the AI about his troubles. He asks the AI, "Why are my soil moisture sensors giving low values on one side of my backyard despite watering the same plants equally each day?". The prompt combined with the real data ensures that the response will be tailored to Bob's issue. With the proposed solution, the AI replies that the cause may be due to different soil composition, different sunlight exposure, or varying drainage for the water. This gives Bob enough knowledge to start investigating the three possible causes to figure out exactly why his plants were not growing at the same rate. This is a classic example of citizen science where people take on the scientific problem for a personal reason. Just like Bob, many others pursuing metrics with LoRaWAN devices would find the AI-driven features on the VISTA+ portal a great help to the citizen science undertaking.

4.3 Approach Outline

To satisfy the goals for the VISTA+ web application, an agile approach that combines research and iterative development was adopted. Initial research included the study of how artificial intelligence has been evolving in recent years, and specifically how it has been adopted in online web applications. Research then transformed into examining approaches to enhance the user experience using AI that will increase user retention, but also genuinely help the user for the purposes they are using the portal. With research in mind, initial prototypes were blackboarded, focusing on ideas that would be compatible with the existing infrastructure of the VISTA portal. These ideas were then confirmed by following the Software Development Life Cycle (SDLC), which is elaborated in Chapter 5. The final stages of the SDLC included an iterative evaluation phase in the form of a user study to determine the effectiveness of the AI solution compared to the data visualization tools in terms of the ability

for a user to analyze their data. The initial feedback from the user study initiated another round of development and research to mitigate some of the early concerns of usability from the participants.

The result following the SDLC was VISTA+, a complete application full of rich AI-driven features and more accessible than ever before. This application can be deployed as a simple update to the VISTA portal, yet it can completely transform the overall user experience of the online application.

Chapter 5

Software Modeling and Implementation

One of the best models to use when creating a complex piece of software is the software development life cycle (SDLC) model [21]. The first few stages of the SDLC include requirement specification and planning. This is followed by architectural design to blueprint the technical needs identified by the initial stages, iteratively. The next stage is implementation, which executes the plans drawn by the architectural stage. The final stage outside of disposal (tearing down software) is the evaluation and testing stage that assesses whether or not the software created is functional and, more importantly, meets the needs identified by the requirements stage. This chapter dives into applying the SDLC model in creating AI-driven features for the VISTA portal to enhance and improve the overall user experience and accessibility.

5.1 Requirements Specification

Requirements specification is arguably the most important stage of the SDLC model. This stage involves brainstorming user stories, which are short and informal statements that outline features as described from the end user's perspective. These statements identify the what and why of features that enable usability in the software. User stories can then be transformed into use cases that serve as the formal building blocks for the software. Finally, use cases are broken down into logical software requirements, which are passed down to developers for architecture and implementa-

tion.

5.1.1 Use Case Modeling

To begin use case modeling for the users of the AI-driven features for the VISTA portal, the intended users must be considered. As discussed in Section 4.2, the two main intended users are researchers and citizen scientists.

User stories for environmental researchers are related to helping them process data faster with some data interpretation capabilities.

1. As a researcher, I want to be able to run frequently made queries fast to save time for the scientific process.

2. As a researcher, I want quick access to my frequently viewed pages to be able to perform my tasks as quickly and efficiently as possible.

3. As a scientist, I want to be able to view my data in terms of a geographical representation to detect patterns caused by geographic elements.

4. As a researcher, I want to have AI assistance in interpreting the data from the data table interface to make valid conclusions from it.

5. As a scientist, I want the ability to accurately predict future values based on existing data using artificial intelligence.

6. As a researcher, I want to have AI assistance in interpreting data from the data graph to analyze significant trends.

User stories for citizen scientists are similar to environmental researchers, but focus more on the user experience and accessibility side of things. For non-expert users, it is essential to think about user-centered design and the proper flow of pages.

1. As a user, I want lots of help in making conclusions from my data so I can solve my citizen-science problems. I also want to be able to ask further questions if I need clarification or am confused.

2. As a user with some visual impairments, I want a textual summary of online content that I might not be able to access with a screen reader.

3. As a super-user, I want to perform frequently completed tasks fast and without much effort (Checking my last 24 hours' worth of data every day, for example).

4. As a user, I want full keyboard accessibility so that I do not have to rely on pointing devices such as mice to use a website.

5. As a user, I want to feel satisfied after using an online resource for my citizen science purpose. This means that I am not constantly under cognitive stress to figure out how something operates.

Combining the user stories into formal building blocks for the software will yield the use cases. Use cases are compartmentalized and specific, they explain how the system interacts with the user in accomplishing their user stories. Table 5.1 outlines some of the use cases for the VISTA+ portal, but they are not a comprehensive list of all the features. These use cases were often revisited in the agile development process to make sure the goals of the portal were met.

5.1.2 Requirements

After crafting the initial use cases, the next step is to define the software requirements. To do this, each use case is broken down even further into logical components that can be implemented as small software changes. Each requirement should have at least a one-to-one connection with a use case. Ideally, the same requirement can serve as a facilitator to multiple use cases. Functional requirements for the VISTA+ portal are represented in Table 5.2 with a priority level assigned to them. A level 1 represents functionality needed for a minimum viable product (MVP), while level 3 requirements serve as a nice-to-have. All the items listed in Table 5.2 were developed as a part of the software, with level 3 requirements being the most difficult to implement.

Tracking which requirements correspond to the specific use cases is an important practice as it allows development efforts to be aligned with the functionality that

Table 5.1: Use cases as derived from proposed user stories for the software.

Use Case	Name	Description
UC01	GraphToText	The user selects the 'AI Analyze' button above and to the right of the data graph in the data visualization page. This gives the user a brief textual summary of the graph they are viewing.
UC02	TableToText	The user selects the 'AI Analyze' button above and to the right of the data table in the data export page. This gives the user a brief textual description of the graph.
UC03	ChatAI	The user utilizes the keyboard to chat with an AI model for data insight. The model answers questions about the user's data in either of the data pages.
UC04	EnhancedFetchData	The user navigates to either data pages and finds popular queries they've performed, ready to select with a button click. The desired data is then fetched and shown to the user in the data pages.
UC05	SuggestiveActions	The user is shown a panel full of suggestive actions on the homepage based on past activity. The user can choose one of those options to and be appropriately navigated.
UC06	KeyboardNavigation	The user can navigate to every page related to data visualization using the keyboard. The flow from each element makes sense in terms of the way the content is presented visually.
UC07	DataToMap	The user can navigate to a page with a map that shows every sensor geographically represented. Clicking on any sensor yields the last available reading from the sensor along with other options.
UC08	VideoClick	The user sees a help icon with the text 'Help Video'. Clicking on the help video pops up a modal with a video on the appropriate topic to help the user perform tasks on the portal.
UC09	MapToText	The user selects the 'AI Analyze' button above and to the right of the map in the map page. This gives the user a brief textual description of the map with respect to the location of the sensors.

Table 5.2: Requirements table for the proposed software.

Req.	Level	Description
R01	1	The system will show an 'AI-Analysis' button on the export data page (tabular data format).
R02	1	The system will show an 'AI-Analysis' button on the data visualization page (graph data format).
R03	1	The system will show an 'AI-Analysis' button on the map page.
R04	1	The system will display a chat interface once any AI-Analysis button is clicked.
R05	1	The system will display an initial response from the AI-Model in the chat interface.
R06	1	The system will display useful information such as average, minimum, maximum, and interesting trends from the readings for the first AI response.
R07	2	The system will allow subsequent chat messages to be sent to the AI model if the user chooses to interface within the chat modal.
R08	1	The system will show frequently visited pages on the home-page after application entry.
R09	2	The system will allow frequently visited pages to be clicked and for the user to be appropriately navigated.
R10	1	The system will show suggested tasks on the homepage after application entry.
R11	3	The system will allow suggested tasks to be clicked and for the user to be navigated to the appropriate page with the task initiated.
R12	2	The system will allow keyboard input on any page within the application for navigation purposes.
R13	1	The system will show all the sensors whose locations are set on the new map page.
R14	1	The system will show a list of all the sensors whose locations are not set on the new map page.
R15	2	The system will let users set a new location or change the location for the sensors through the map page.
R16	1	The system will show users the last reading from the sensor if the sensor is clicked on the map page.
R17	3	The system will display geographical factors related to sensors when the 'AI Analysis' button is pressed on the map page.
R18	1	The system will display multiple help video buttons with text and tooltips labeled with 'Help Video'.
R19	1	The system will load and play help videos in a modal when the user clicks on the help video buttons.

Table 5.3: Requirements traceability matrix mapping each use case to the requirements for software.

	UC01	UC02	UC03	UC04	UC05	UC06	UC07	UC08	UC09
R01		X	X						
R02	X		X						
R03			X						
R04	X	X	X						X
R05	X	X	X						X
R06	X	X							X
R07	X	X	X						X
R08					X				
R09					X				
R10					X				
R11				X	X				
R12						X			
R13							X		
R14							X		
R15							X		
R16							X		
R17			X						X
R18								X	
R19								X	

each use case demands. This makes it easier to prioritize work and forces developers to revisit the main goals for which the software is being implemented. A way to efficiently map use cases and requirements is a requirements traceability matrix, such as the one shown in Table 5.3. From the table, it is evident that Use Cases 01, 02, 03, and 07 are highly dependent on some of the same requirements, so it was beneficial to begin development by prioritizing those requirements. Overall, the requirements traceability matrix helped estimate the development effort required for each use case. The more complex use cases were tackled early on in the implementation to ensure that they would be complete for the project timeline.

Several non-functional requirements were also implemented with the functional requirements, as they highly contribute to the overall user experience of the portal. The list of non-functional requirements is highlighted below:

1. The data visualization tools should load in less than 3 seconds for smaller pe-

riods (1 hour through 7 days).

2. The initial AI response should load in less than 15 seconds, regardless of the period selected by the user.

3. The application and specifically the data visualization tools should be responsive in screen size to accommodate use by mobile and tablet devices.

4. Any conversational prompts sent to the user after the initial prompt should load in less than 10 seconds.

5. The changes from this work should be deployable via a simple software update that is typically done to build new features or fix bugs.

6. There should be a backup of user event data in case of data loss or corruption in the original source.

7. Suggestive actions should only involve a single click from the user, making it as minimal an effort as possible. The task should be fully executed with one click from the suggested actions menu.

5.2 Software Architecture

The VISTA+ portal has many different components that must work in tandem to provide a robust and comprehensive experience to its users. The architecture includes a backend and a frontend that communicate via a RESTful API. The backend provided by The Things Stack (TTS) is also essential for optimal functionality for the VISTA+ portal. There are also several cloud computing technologies implemented as a part of the new functionalities introduced in the VISTA+ portal. The section focuses on the architecture, showing the relationships between all the subsystems and describing the design process that led to decisions for the implementation. A comprehensive overview of the new software architecture is given in Figure 5.1. The new changes compared to the architecture for the VISTA portal are circled in red.

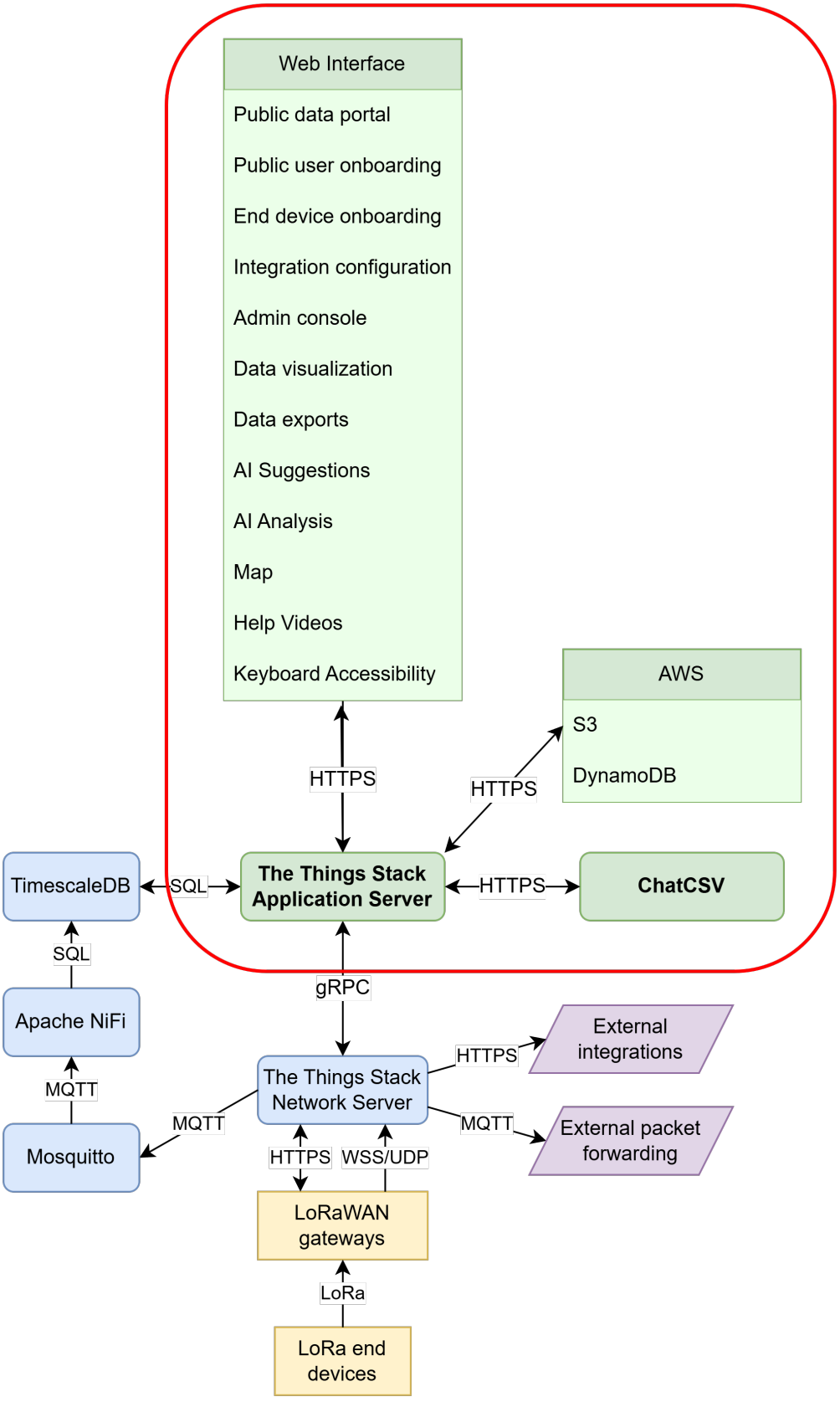

Figure 5.1: New architecture displaying the changes for the VISTA+ portal.

5.2.1 Flask Back End

The backend for the original VISTA portal was built using a Flask App in Python. The Flask framework exposes its functionality through a RESTful API. This allows other applications (the front end) to interact with the Flask app by making HTTP requests to predefined endpoints. This custom backend contains APIs for the existing functionality for fetching data to be displayed in the front-end. This backend now also contains all the new features as part of this thesis. Whenever the Flask app is mentioned, this is the back end being referred to.

5.2.2 The Things Stack

As greatly detailed in Section 2.3, the Things Stack serves as the starting point for the software developed for the VISTA portal. This same tech stack is used to further host the changes that are being introduced as a part of this thesis. The Things Stack includes a backend and a front end made by The Things Industries. Their backend will not be modified, but the front end (made with the React front-end framework) will be heavily modified to establish the latest features from this thesis.

5.2.3 ChatCSV

One of the first features worked on was the ability to get a textual summarization of data through an AI model. This feature alone would take care of Use Cases 01, 02, and 07. The initial design for this feature was to upload CSV data to one of the readily available LLMs, such as Google's Gemini, OpenAI's ChatGPT, or DeepSeek. The idea was to use their Application Programming Interfaces (APIs) to send the CSV data along with a short prompt asking the model to summarize key findings. Experimentation was done with all 3 of the models aforementioned to test response time, quality of the response, and total cost. This process would at least perform a proof of concept for this feature.

Google's Gemini was unable to take a comma-separated value (CSV) file of the data, which meant that the data would either have to be processed into a large stream

of text or somehow converted into a different format that would allow the model to analyze it [9]. Taking this setback, DeepSeek was tested next to see how it would perform overall.

DeepSeek was able to process some CSVs with decent performance [10]. While testing this model, the server remained busy several times when trying to upload the CSV of data, despite the file size being well under the limit specified on the website hosting the model. Still, DeepSeek remained an option since it was able to process the data and offered some low-cost options once the free threshold was hit. With the open-source nature of DeepSeek, self-hosting the model was also an option, but the implementation and management would have been outside the scope of this thesis.

ChatGPT offered promising results, being able to process relatively large CSVs with over 10,000 rows of sensor data without much issue [14]. The responses generated a summarization of the data that was appropriate to use in the VISTA+ portal. Along with summarization, the model was able to handle further querying the data and asking it questions such as 'what was the highest recorded temperature in April' or something more complex, such as 'What was the time series autocorrelation of temperature'. ChatGPT seemed to be the best option out of the models examined during this process. The only downside was the cost of using the API to get the same responses given similar input. With the same size of input (CSV file with over 10,000 rows plus a short prompt), the cost of using GPT-4-Turbo with the API would be $0.01 per 1,000 tokens of input and $0.03 per 1,000 tokens of output making the total cost of the questions approximately $0.04 - $0.06. While this amount isn't very significant, it would have accumulated to a substantial amount if put in production or even while testing and evaluating during this thesis.

At the end of the process, a proof of concept was established, backing the idea that this feature was possible and could be implemented even if there was a cost involved for using artificial intelligence technology. As a last attempt to find a resource that could perform as well as ChatGPT yet remain cost-effective, an online search was done with the phrase 'CSV AI Tools'. Among the tools found with the search was a

tool called ChatCSV ([8]).

ChatCSV is a free tool that incorporates AI to help analyze CSV files. It allows users to utilize their online interface to upload a CSV file and then query it using natural language-exactly the solution needed for VISTA+. Even more significant was that in the background, ChatCSV is using OpenAI's ChatGPT models for its functionality. Upon further research, it was found that ChatCSV also had an API that was free to use since their acquisition from a company called Flatfile. With the API, developers can currently choose from OpenAI's GPT-4 or GPT-3.5 Turbo models, all without cost and limitations. An example of the API format for ChatCSV's API is given in Listing 1. The API is authorized via a token generated on the platform. The model is chosen at the API level, and messages are provided as an array comprising two roles, user and assistant. Though the use of ChatCSV warranted more development work to convert sensor data into a publicly available CSV file. This introduced cloud computing technologies to the architecture of the project.

5.2.4 AWS S3

The requirement of ChatCSV's API for the CSV file to be accessed by a publicly accessible link introduced the need for Amazon Web Services' (AWS) Simple Storage Service (S3). S3 is a popular cloud object storage service by AWS that allows file uploads and access using their readily available API or user interface. Being an enterprise service, there are costs involved for using S3, but the threshold is easily reset by deleting unused files from the storage service.

A bucket (name of the individual storage unit) was created with the name 'csv-bucket-nikhil' to start uploading the CSV and generating the publicly accessible link. The bucket was configured to make the objects within it accessible with a public link. Here are the steps taken to store a CSV file of the data requested from VISTA+'s end user.

1. The user interfaces with the front end to obtain data on either of the data visualization pages (tabular and graph). This sends a request to the back end

```
{
    'Authorization: Bearer <token>'
    '{
        "model": "gpt-3.5-turbo-16k-0613",
        "messages": [
            {
                "role": "user",
                "content": "What is your name?"
            },
            {
                "role": "assistant",
                "content": "bob"
            },
            {
                "role": "user",
                "content": "Tell me interesting details
                    about this data. Summarize the data
                    for me as well."
            }
        ],
        "files": [
            "Some public file link"
        ]
    }'
    'https://www.chatcsv.co/api/v1/chat'
}
```

Listing 1: Example format of the ChatCSV API.

with the parameters that the user has selected.

2. The return of data to the front end does not change. Instead, the same function responsible for returning that data performs a few extra steps. It converts the data to a Pandas data frame, enabling it to be written as a CSV file.

3. Next, the converted CSV is uploaded to the above-mentioned S3 bucket. The CSV file is named with respect to the timestamp at which the file is uploaded. Then a static link is generated with this format:

   `s3_bucket_name.s3.amazonaws.com/sensor_data_{timestamp}`

4. The static link to the CSV is then returned to the front end.

The AWS S3 service and the same process above were also used to store CSV files full of user event data for the AI suggestions feature. This is later discussed in Subsection 5.2.6 below.

5.2.5 Invoking the ChatCSV API

Upon receiving the public CSV file link from the back end, the process to generate the initial LLM response begins. At this point in the process, the user can now see the data visualization tool populated with the data. In the background, another request is sent to the Flask back end to fetch the initial response from ChatCSV. In this request to the back end, a prompt must be given to receive a desired response, which would be useful to the end user. Prompt engineering was the first great challenge that occurred within the implementation of the software and is discussed in detail in Subsection 5.3.1. After the request to the Flask back end was sent, the back end invoked the ChatCSV API that would eventually return a string containing the response, which was then sent back to the front end to be parsed and displayed.

The initial implementation of Use Cases 01, 02, and 09 stopped at generating a summarization of the data and displaying it to the user in the appropriate data visualization format. After showing the prototype to potential stakeholders, the ability to converse with the large language model was born. This led to changes in the

back-end and front-end implementation to allow a steady stream of messages to be executed between the AI assistant and the end user.

5.2.6 Suggestive Actions

Use Case 05 is about allowing users to view suggested actions and being able to execute those tasks with ease. The software implementation for this feature required rigorous research and multiple stages of development. The architecture was brainstormed and iterated on upon learning new facts during research.

Stage One: To be able to suggest any tasks or navigational items, user activity would have to be tracked. There were multiple ways to do this, but given the recent addition of cloud computing services via Amazon Web Services (AWS), it was decided that user events and activity would be stored in another AWS service or at least another cloud computing provider. The architecture needed a cloud service in the form of a database, and options were narrowed down to AWS' Relational Database Service (RDS), SimpleDB, and DynamoDB.

- **RDS**: The Relational Database Service is simply a cloud-hosted SQL-based database service that allows for provisioning, configuring, backing up, and patching to be done with ease with AWS [4]. While super powerful, it was decided that RDS would not be used for the VISTA+ portal, given that the data being stored would be inconsistent and unstructured, which is suboptimal for a relational database.

- **SimpleDB**: SimpleDB was the next option in the list and was eliminated after comparison with DynamoDB. SimpleDB is a NoSQL solution that could have very well been used for VISTA+, as it offers the flexibility of inconsistent data input. Though this technology is not able to scale and ends up automatically indexing every attribute, making it query flexible at the cost of performance. DynamoDB, however, was slightly different than SimpleDB and able to support the needs of the database perfectly.

Table 5.4: Example structure of the user events table in DynamoDB

user_id	time	event_desc	event_type	extra	page	project
admin	20250408_190414	Fetch Data clicked on...	FetchData	{...}	expdata	proj1
user3	20250408_203508	Navigated to the Data...	Navigation	null	datavis	proj7
admin	20250408_235327	Navigated to the Map...	Navigation	null	null	proj1
user3	20250411_133419	Fetch Data clicked on...	FetchData	{...}	datavis	proj1

- **DynamoDB**: DynamoDB is a serverless NoSQL database solution that partitions the database with a primary attribute and an optional sorting key if specified during database creation. This structure matched the need for the user events table, where the primary attribute is the user ID associated with each new user on the VISTA+ portal, and the optional sorting key is the timestamp so that the table can be efficiently queried based on the timestamp. Serverless technology in general is an emerging technology that has gained popularity for its lightweight and management simplicity [12]. In this case, it was the perfect solution for a simple database hosted in the cloud.

Having chosen DynamoDB for the database to store user events, a new table called 'nikhil-user-events' was created with the partition key 'user_id' and the sort key 'time'. Using AWS' official software development kit (SDK), boto3, an API was created in the flask app that sent user events to the table. The rest of the table attributes and structure are demonstrated in Table 5.4.

Stage Two: Now that user events were being logged and stored, the method to predict next user events and suggest those to the user had to be created. ChatCSV was leveraged once again to complete this bit of implementation. A test was performed by navigating the portal and completing various tasks on the platform from the perspective of both the researcher and the citizen-scientist. Then ChatCSV was used

to get 4 suggested actions for each of the two users based on the user events that were logged, and the response was promising. After more experimentation to reach a proof of concept, development began to integrate this into the VISTA+ portal. An API was created in the Flask APP with an input parameter of the user ID coming from the front end. A query using boto3 was performed to fetch the most recent 100 user events associated with the inputted user ID and sorted by time. These were then converted to a Pandas data frame, transformed into a CSV file, and uploaded to another S3 bucket with a publicly accessible link. From here, the ChatCSV API was used to get the user suggestions in a consistent JSON format so that they could be parsed and somehow programmed to be executed in the front end. This process eventually led to the challenge of stochastic responses from artificial intelligence and is detailed in Subsection 5.3.3.

Stage Three: Now that user suggestions were being generated, the only thing left to do was to figure out how to get the application to perform the task. This task was carried out via URL-based state management. The response from the ChatCSV API included a JSON object with the necessary information to carry out tasks for several pages. The tasks were limited to two types: navigation and fetching data from the data table and the data graph. The page associated with the task was navigated to using the React Router Dom library, which enables dynamic routing in React applications. If the task was of 'FetchData' type, significant data would be taken from the JSON and carefully placed into the navigation link using query parameters. Query parameters allow for data to be sent to different pages without having to rely on a data store or further asynchronous processing on the target page. Once the navigation is complete, a 'useEffect' React hook is utilized to prepopulate form input and trigger the fetch data API, showing users their chosen data visualization.

All three stages combined made it so that one click from the user would either navigate them to the correct page or automatically perform all the steps needed to run the fetch data function on the Export Data and Data Visualization pages. Figure 5.2 is a diagram depicting the processes and data flow for the suggested actions feature.

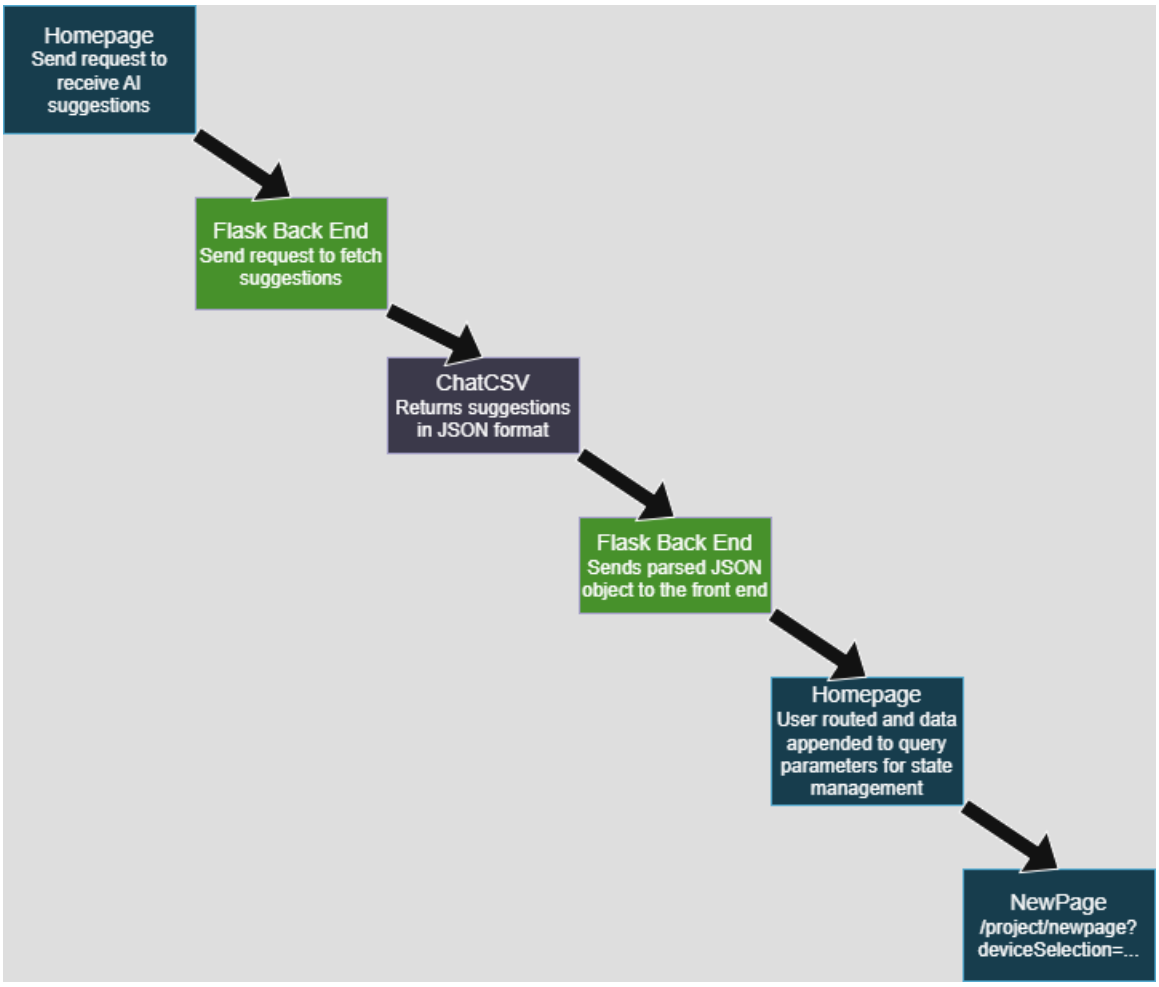

Figure 5.2: Diagram of processes for Suggestive Actions.

Use Case 04 was implemented following the same approach for suggestive actions by filtering out the suggestions to be type 'FetchData' and displaying them on the appropriate page.

5.2.7 Geographic Sensor Representation

The map page (Use Cases 07 and 09) was developed in two stages. The primary stage sets up the core front end needed for the new features. The secondary stage added the ability to analyze the data concerning geographic sensor placement using an artificial intelligence large language model.

The primary stage included accumulating individual sensor locations, displaying them on a map, and creating an API in the Flask app to fetch the last reading from each sensor on the page. As part of the base functionality provided by The Things Stack Open Source edition, users can set individual sensor locations and view each separately on a map. The changes here amalgamate the locations onto one page so that the relationship between the sensors can be studied. The front end for this feature was put together by using the Leaflet library [2]. Markers were used to represent each device, and clicking on the marker would show the last reading and a few other device management options. For devices whose locations were not set, the user interface displayed the device details on a card-like interface with the option to set the location of the sensor.

The secondary stage took inspiration from the AI-analysis features on the 'Export Data' and 'Data Visualization' pages. Device information, last reading, and geographical coordinates were converted to a CSV file and fed into the ChatCSV API for further analysis using the ChatGPT LLM. The initial prompt asked the model to predict the conditions in that geographical region and associate those with the values from the sensors. The user can further query the model after the initial response is loaded.

5.2.8 Supplementary Features

The implementation of Use Cases 06 and 08 will be discussed in this subsection. While these features are not implemented using artificial intelligence, they are supplementary features that aim to increase the usability of the application by making it more accessible and offering help to new users.

Once the main features for the VISTA+ portal were complete, it was time to start evaluating usability from the perspective of a new user on the platform. One of the main things noticed was that the new users to the portal might be confused by the verbiage, and how that might cause confusion as to how to use the portal. To remedy this issue, help videos were recorded for all the major portions of the portal. The content of the videos includes screen captures and members of the team talking about the significant elements on the page. These videos were then placed in modals for them to be accessed by the user by finding buttons with a help icon or with buttons saying 'Help Video'. This feature intends to help users get used to the verbiage used in the portal (things like 'projects' and 'gateways').

The other supplementary feature is targeted specifically towards increasing accessibility for the disabled and ease of use by superusers. The pages on the portal were audited for keyboard accessibility and the general flow of elements. In areas where, while using the keyboard for interacting with user interface elements, the flow was disturbed by skipping elements or awkward behavior, the page was reformatted and examined to fix the flow of interaction. Most pages were found to be okay with only slight changes needed. Three pages were found to be completely inaccessible, so that it was impossible to interact with one or more elements on the page. An example of that was the 'Data Visualization' page, where the 'Aggregate By' input was being completely skipped while using the keyboard to fill the form. Major pages where the user was most likely to spend greater time were audited and improved to be more keyboard accessible.

5.3 Challenges

This section speaks to the challenges experienced by developers during the development and software modeling of the VISTA+ portal. The specific challenges are discussed along with a review of the literature on how these challenges have been remedied in related works.

5.3.1 Prompt Engineering

Prompt engineering is an entire field of computer science within artificial intelligence, and its significance was quickly established during the creation of prompts for the VISTA+ portal. Prompts are "instructions given to an LLM to enforce rules, automate processes, and ensure specific qualities (and quantities) of generated output" [23]. White et Al in [23] discuss the five categories of prompt patterns that must be considered when writing a prompt. Of the sixteen specific prompt patterns mentioned in [23], the Context Manager and Output Automater patterns were the most applicable to the type of prompts intended for the portal. The Context Manager pattern intends to focus the conversation on specific topics or exclude unrelated topics from consideration [23]. This pattern was utilized to focus the AI conversation towards environmental monitoring, giving the user a better experience using the LLM for data interpretation. The Output Automater was specifically helpful during the development of suggestive actions, where the output must always contain JSON that enables the automatic navigation and task execution in the front end to work properly.

The challenge for the portal was mostly related to the initial prompt sent to the LLM to get a response with enough context that it would be useful to the user. All further interactions between the user and the LLM through the chat interface would be automatically tailored to the initial context given to the model. Eventually, it was solved through iterative adjustments to the verbiage using the patterns described in [23]. Table 5.5 highlights the different prompts and their points of failure for providing a response that would be useful to the user.

Table 5.5: Prompts and the associated challenges encountered in the response from the LLM.

Prompt	Problems in Response from LLM
'Tell me interesting details about the data. Summarize the data for me'	Response would return visuals and reference each column with the raw name. For example, referring to soil moisture as 'water_SOIL'.
'No visuals. Here is a map that associates raw column names to the human-readable column name: {water_SOIL: Soil Moisture, ...}. Summarize the data for me,	No context towards environmental monitoring and still mentioning raw elements such as 'dev_eui' (sensor identification number). Assumes that all sensors can record all values.
'This CSV is related to sensors recording environmental data. Tell me interesting details about the data. No visuals. Only use the display names to refer to a column. Here is the mapping for column headers and 'dev_eui's: {someDeviceID: {water_SOIL: Soil Moisture, ...}}. Summarize the data for me in a simple way.'	The model has a hard time calculating mathematical summaries as it responds with a statement related to being unable to perform statistical analysis for NaN values. This likely means it's trying to read columns where the data is null or doesn't exist.
'Tell me interesting details about this environmental monitoring data. No visuals. Only use the display names to refer to a column. Each unique 'dev_eui' represents a unique sensor, and so if a column is missing data for a 'dev_eui' that probably means that sensor doesn't record that value. Here is the mapping for column headers and 'dev_eui's: {someDeviceID: {water_SOIL: Soil Moisture, ...}}. Summarize the data for me in a simple way.'	Very close to the ideal response. A few further tweaks in verbiage finalized this prompt to be used for the 'Export Data' page AI-Analysis feature.

This process was repeated in the places in the application where the AI-Analysis feature was added. To maintain an aesthetic appeal to the end user, the initial complex and long prompts were replaced with the text, 'Tell me interesting details about the data. Summarize the data for me', so as not to overwhelm the user.

5.3.2 Data Pipeline

Throughout the development and evaluation of the VISTA+ portal, a real-time data pipeline had to be maintained to test the constant changes in software. The data's final destination, a Timescale database managed at the University of Nevada, Reno, was hard to access as a direct tunnel to the database was not possible if developing outside the university network. This made software implementation a challenge since data could not always be pulled in unless the development was happening within the network at the University. Evaluation of the software (discussed in detail in Chapter 7) would have been more difficult without real-time access to data. Eventually, two different methods to solve this challenge were implemented.

1. **Replicated Data**: Real data from sensors deployed at one of the developers' houses was replicated into a locally running Timescale instance. While not a long-term solution, this made sure that the developers had access to at least some data so that software could be adequately tested. In the early stages of development, this data was often replaced so that AI responses could be tested with a diverse data pool.

2. **Virtual Private Network**: As a long-term solution, a virtual private network (VPN) was set up on the server hosting the Timescale database. This was done via a tool called Tailscale, which makes it so that devices independent of their physical network can be on the same network as the host machine. This allowed the developers access to real-time data regardless of the location from which they were working.

5.3.3 Stochastic Responses

Another great challenge in the software implementation process was stochastic responses from the AI. A stochastic response means that given the same input to the LLM, the end response is different in multiple runs. LLMs work by trying to predict the next word/token in the response, and certain randomness is introduced to have the model feel natural in conversation. Sometimes this randomness is turned off, and responses are more deterministic (the same input gives the same response multiple times). The suggestive actions feature relies heavily on the JSON response from the LLM so that it can be correctly parsed and used to execute the tasks for the user. Even with a strong prompt telling the LLM to return only valid JSON, it was observed that the response varied too often. This made it difficult to write a valid Python script that could extract the JSON when the response contained too many random elements in every iteration of the response.

The standards for deterministic AI responses are generally low. Many papers, [5] for example, discuss that non-determinism can be problematic with LLMs because it can create hallucinations (inconsistent and sometimes incorrect AI responses). This is a grave problem for safety-critical systems, and it is generally recommended that practitioners need to "scope the unpredictable effects artificial intelligence and machine learning may have on decision-making and system control" [5]. While the VISTA+ portal isn't necessarily a safety-critical system, maintaining a level of consistency with AI responses sets the standard for user experience in the entire portal. Users are unlikely to use and trust a tool that is too inconsistent too frequently. Specifically, with the absolute need for a consistent JSON response for the suggestive actions, it was imperative to find a solution to this issue.

This challenge was overcome by studying the response and finding common patterns that could be programmed into a script. In responses where the LLM returned exactly what was expected (a stringified JSON object), no changes were made, and this response was returned to the front end. In responses where the LLM returned

additional details along with a valid JSON object, the extra details were filtered out of the response, and the final JSON object was returned to the front end. In responses where the LLM completely failed to send any JSON object or sent a JSON object that was malformed, the ChatCSV API was re-executed a maximum of three times to obtain a valid JSON object. If after the three attempts there was no luck with the LLM, an error popped up to the user, and they were told that this feature is currently under maintenance.

Chapter 6

Application Scenarios

This chapter begins by reviewing new components of the user interface introduced as a part of this thesis. All major UI and applicable features are discussed here. The latter half of the chapter discusses possible scenarios of use within and outside of environmental monitoring.

6.1 Interface Overview

The final product has many different components that work in tandem to satisfy all the requirements and usability goals mentioned in Chapter 5. The general user interface design has been maintained since the VISTA+ portal was built on top of the existing portal for environmental monitoring.

Figures 6.1 and 6.2 represent the suggestive actions feature. Once the user logs in to the portal, they are routed to the homepage with the modal already active. After a few seconds of loading time, the suggested actions are shown. The top two suggestions shown in the modal are ones where they are performing a data task on either of the applicable pages. After clicking either of the tasks, the user is navigated to the appropriate page with the task already started for them. Figure 6.2 demonstrates a task already started and in the loading state on the data visualization page following a click on the suggested action. The bottom two suggestions shown in Figure 6.1 are for navigation, where the user is simply routed to the page described in the action.

Figures 6.3 and 6.4 are examples of the AI analysis features on the data export

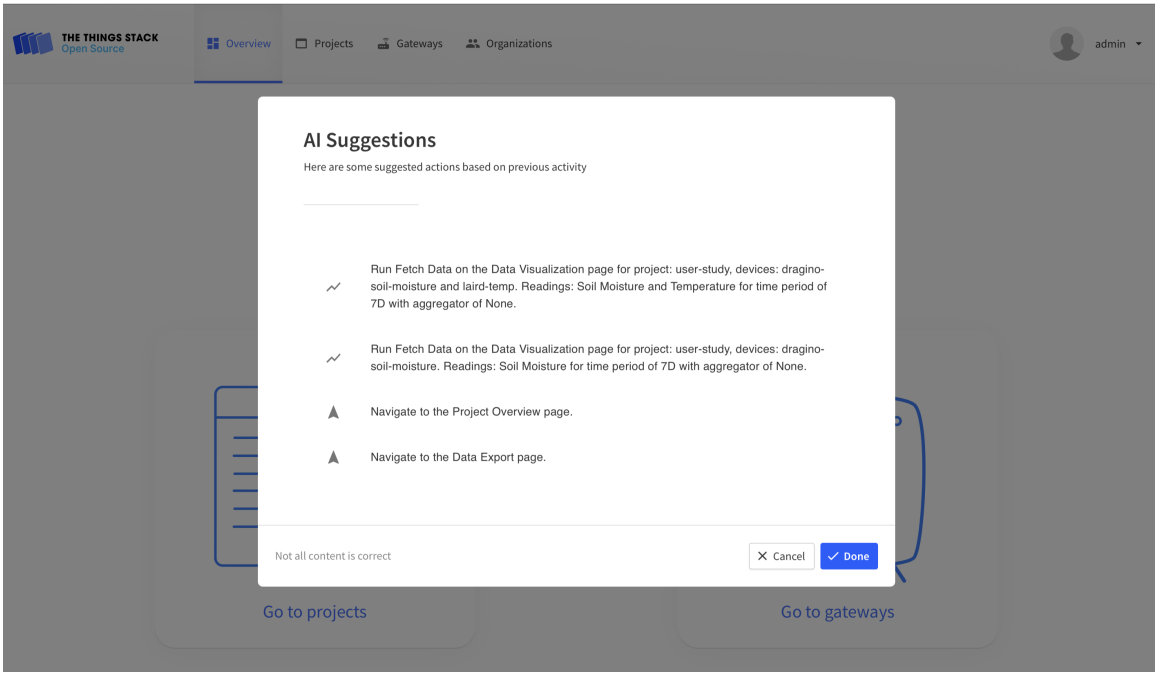

Figure 6.1: Homepage of the VISTA+ portal showing user suggestions.

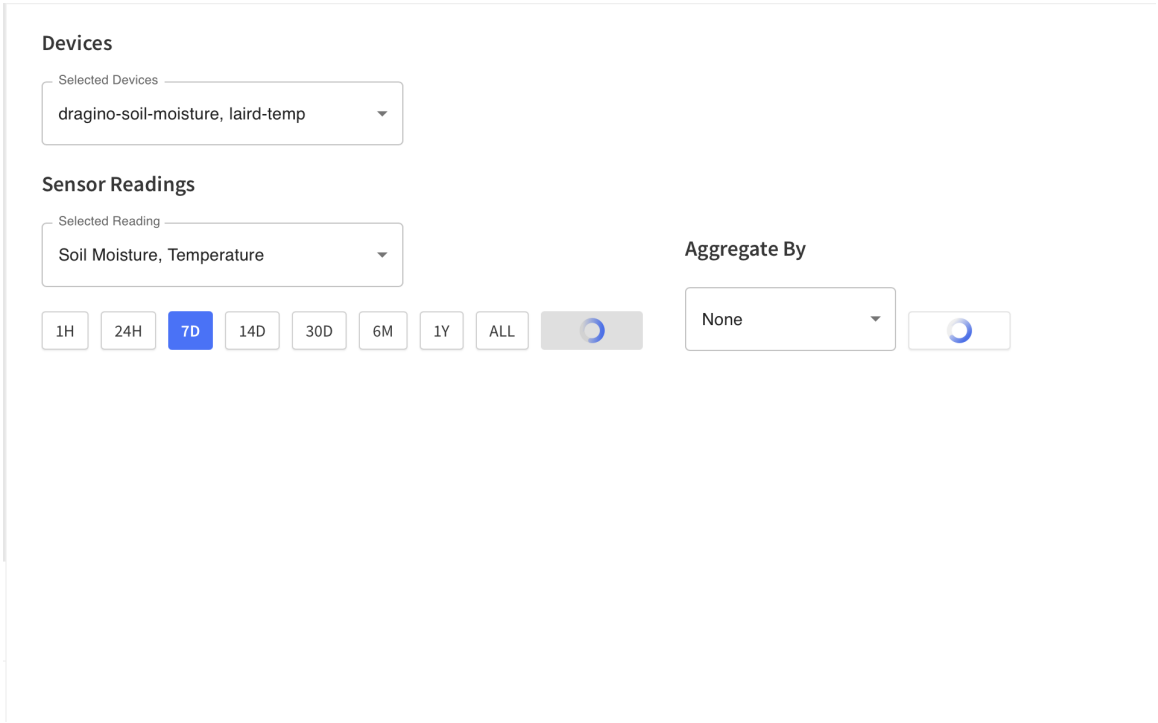

Figure 6.2: Loading state on the data visualization page after user action click from the homepage.

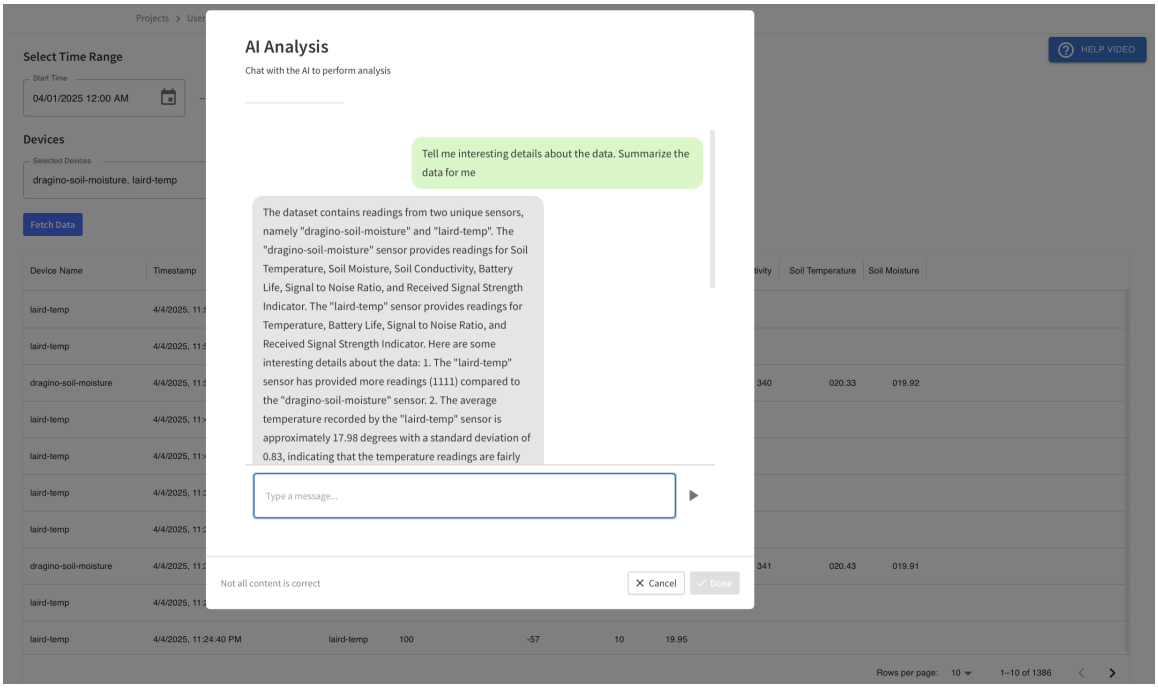

Figure 6.3: AI Analysis feature on the export data page.

page. The data export page displays a tabular form of data after the user has selected the parameters (devices and timeframe). Once the data has loaded, users can export the data in various forms. The AI response is targeted towards being able to provide summaries of the data with basic statistical analysis, such as an average of values and outliers, along with any interesting takeaways that the AI might be able to pick up on. Figure 6.4 demonstrates the platform's capability of generating a graph from the data on the data export page. If the user decides to click on the graph, it is opened in another page in its original size.

Figures 6.5 and 6.6 are samples of the AI analysis feature on the data visualization page. The page without any enhancements allows users to select the devices, readings, and a time frame for which they want to view a graph. The large language model response on this page is targeted towards mimicking the trends of the graph via a textual description. The prompt to the AI is made specifically to have the textual description describe the contents of the graph so that it could be interpreted via screen readers. The chat interface colors messages from the AI assistant in grey while

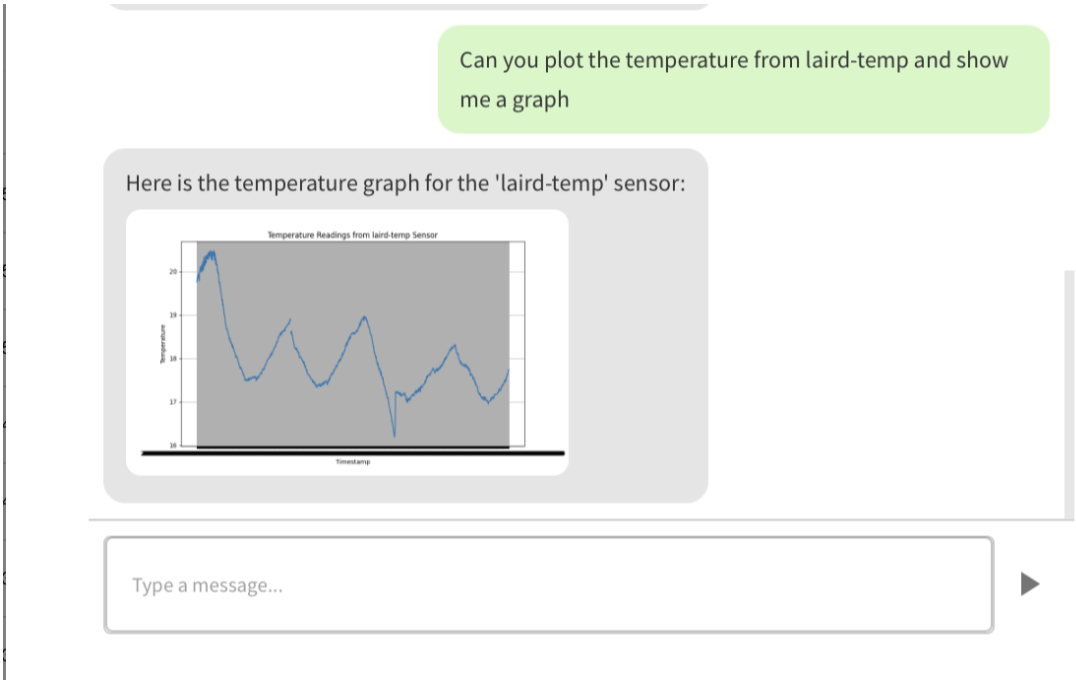

Figure 6.4: Chat interface where the AI response includes a graph of data.

keeping user messages green. Below the active chats, a text input and send button enable the user to send more messages to the assistant for additional help. Similar to the export data page, this page and chat interface also allow for further querying of the model as seen in Figure 6.6.

Figure 6.7 is an image of the new map page on the VISTA+ portal. As mentioned before, existing functionality on The Things Stack portal allowed users to add individual sensor locations and view that single location on a map (Figure 2.3). This new page accumulates all device locations and displays them on a map with other helpful information. Once a sensor is clicked on the map, a small card containing the type of sensor, last reading information, and a couple of sensor management options is shown right above the marker. For unmarked sensors whose location has not been set by the user, an individual card is displayed at the bottom of the page that displays the sensor type, last reading information, and an option to set the sensor location. Once a location is set for an unmarked sensor, the card representing the sensor on the bottom of the page is replaced with a marker on the page.

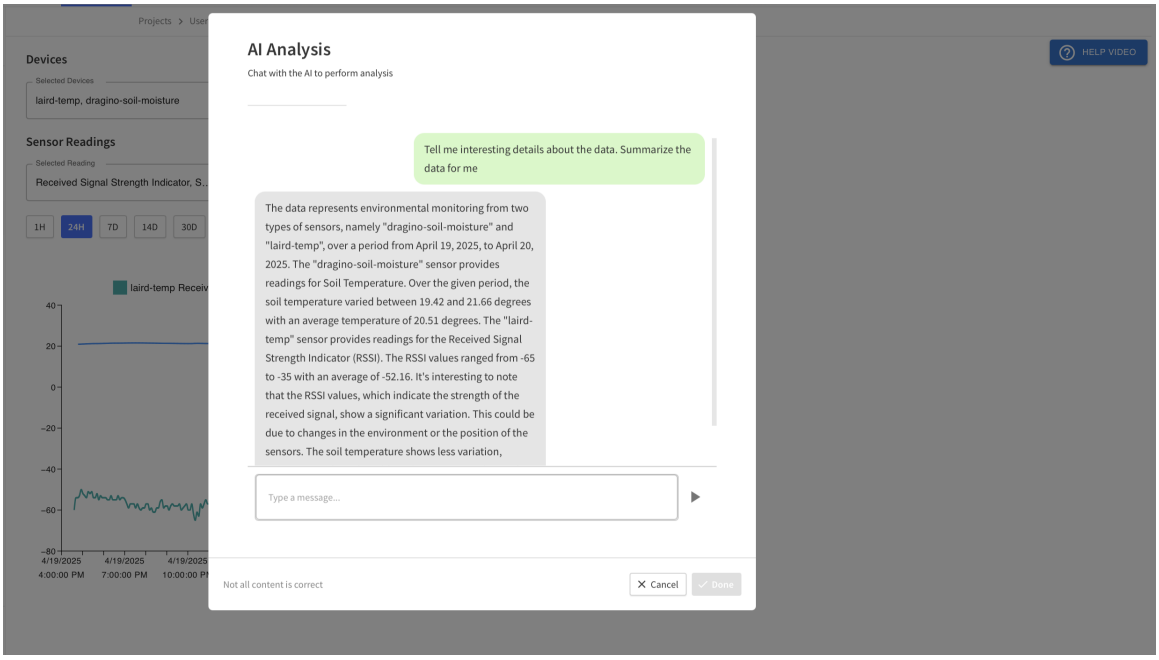

Figure 6.5: AI Analysis feature on the data visualization page.

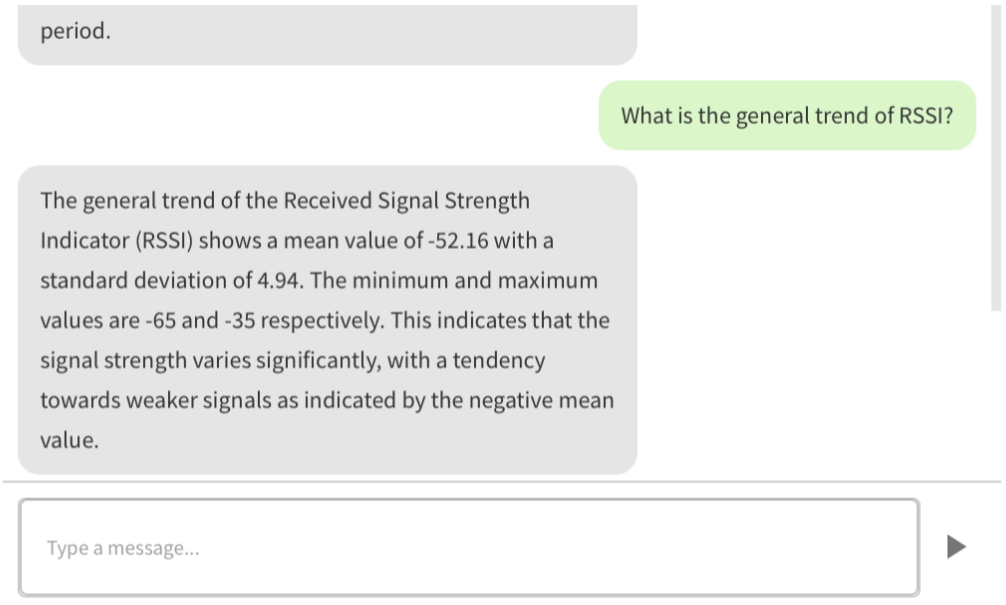

Figure 6.6: Chat interface where users can further query the model for a deeper analysis.

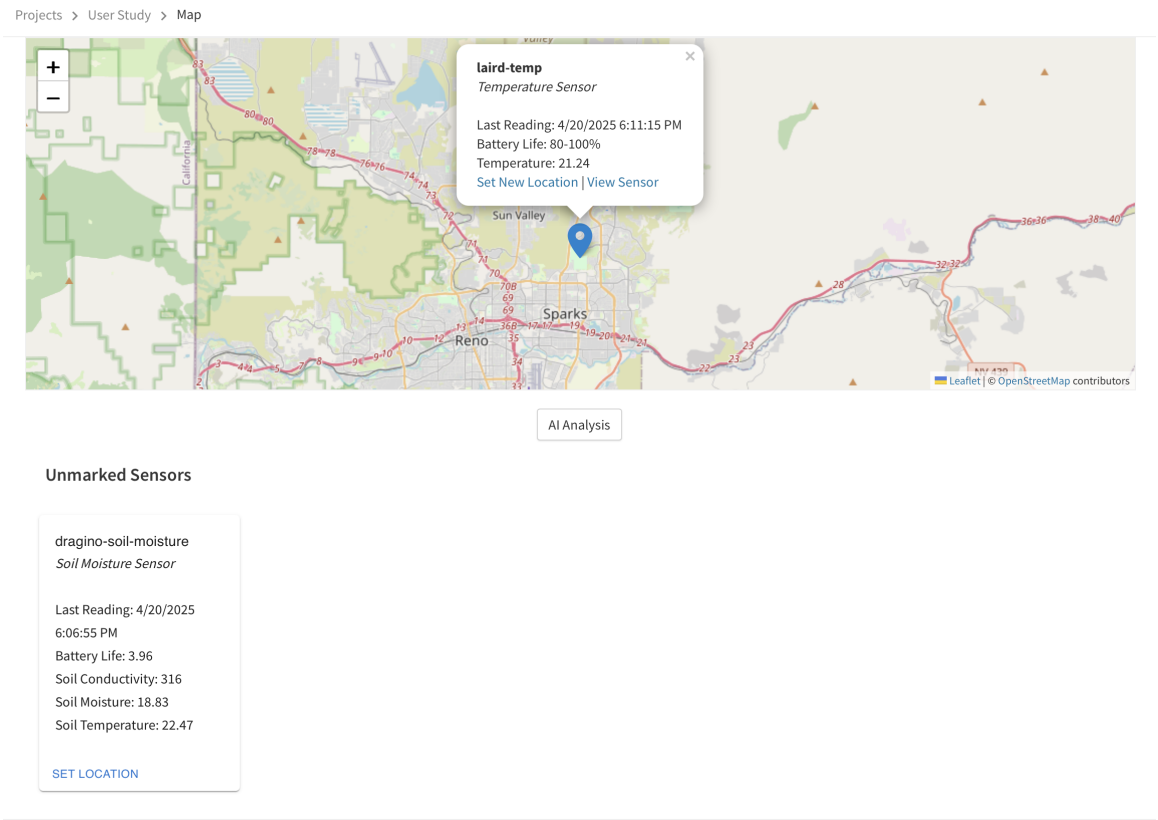

Figure 6.7: New map page on the VISTA+ portal with a sensor on the map.

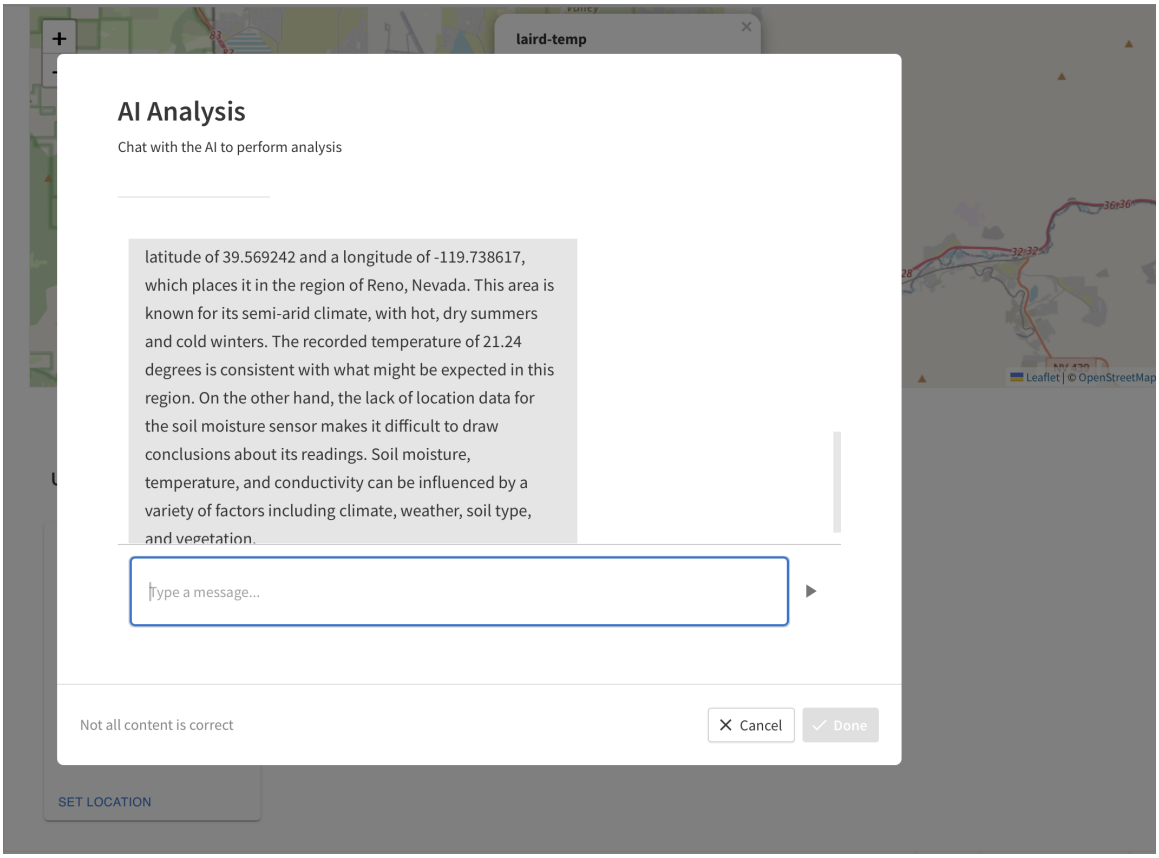

Figure 6.8: Chat interface on the map page with response related to geographic information.

An AI analysis tool was also added to the map page, and it is shown in Figure 6.8. This tool works similarly to the others except that the response is focused on creating meaning between the location of the sensors and the readings they are giving. The initial prompt given to this instance of AI usage differs heavily from the other places where a similar feature is introduced. The data associated with this page is limited to the last reading of the sensors on a particular project, along with the longitudinal and latitudinal position of the sensor. The ChatCSV API is then fed this information in CSV format and asked to provide a summary of the map and any meaningful connection between sensor value and location. For example, in Figure 6.8, the developers intentionally set the location of this sensor to Reno, Nevada, and the AI was able to study past climates to inform the end user that the recorded temperature of "21.24 degrees is consistent with what might be expected in this region."

Figure 6.9: Example help video buttons that trigger help video modals.

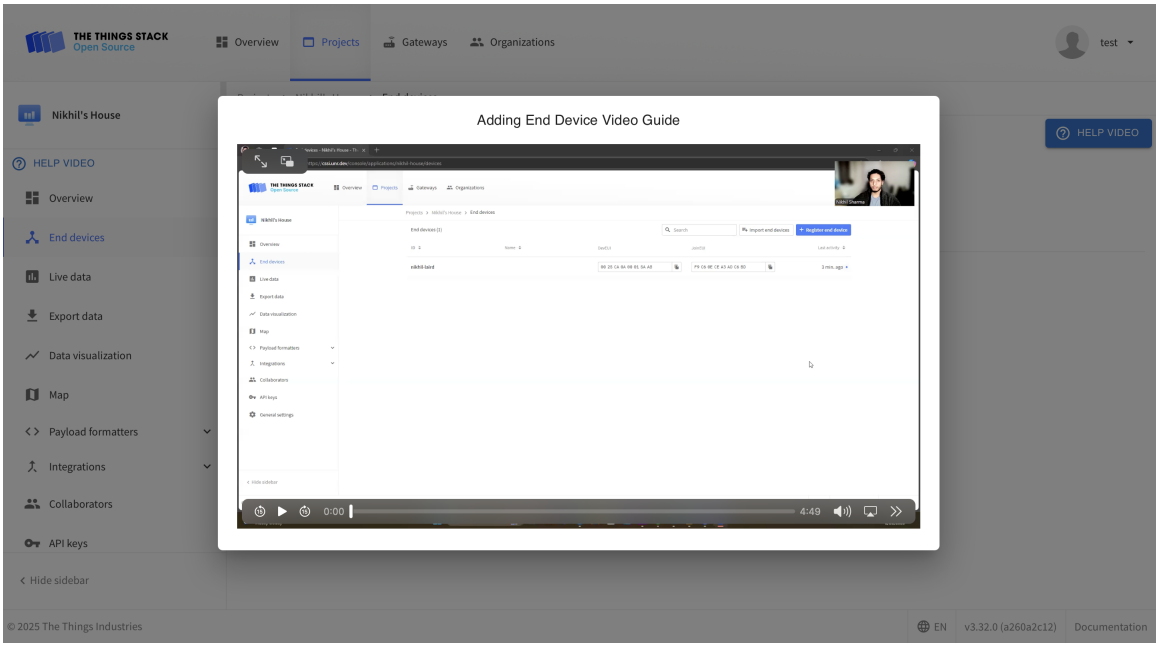

Figure 6.10: Help video modal on the add end device page.

As a part of Use Case 8 from Subsection 5.1.2, videos were added to help users get familiar with using the portal. Buttons similar to the buttons in Figure 6.9 were placed in various parts of the online application so that the videos could be conveniently accessed by users. A total of 8 videos were recorded, showcasing different features and informing users about the more complex pages and their functions. The videos included the developers talking over screen recordings to explain the various parts. Figure 6.10 is an example of an active video modal shown after the help video button is pressed on the add end device page.

Other use cases and functionality implemented as a part of this thesis cannot

be pictured. These features include improvements to keyboard accessibility and the general flow of logic in front-end elements.

6.2 Scenario A: Commercial Applications

6.2.1 Retail Intelligence

Artificial Intelligence is already making its mark on the online retail and shopping industries. These trades have found a growth in AI-driven chatbots that have emerged to help consumers obtain personalized customer support [16]. This functionality could be extended to provide a tailored shopping experience using data-based suggestions and support. User activity could easily be tracked to log the items that the user is spending more time looking at and then fed into an AI model to return custom suggestions from a database of items sold by the retail company. A chatbot specifically loaded with these suggestions could also be presented to the user to shop solely using natural conversation. The intersection of AI and online retail could provide endless opportunities to increase the user experience of retail websites.

6.2.2 Financial Management

Financial management institutions could highly benefit from the work presented here. Financial data is often very diverse, with too many complex data models coupled together. Navigating the complex environment might be difficult for beginners in this industry, and artificial intelligence-driven tools can help. AI summarization of graphs and data similar to those presented in this thesis could instantly boost the productivity of wealth managers and the institutions for which they work. With slight modifications and clever data filtering to prevent sensitive data leakage, an AI tool could also be used to make intelligent market predictions that staff could verify. Having more time to make critical decisions for placing trades or scheduling investment orders is highly appealing for this industry.

6.3 Scenario B: Essential Services

6.3.1 Smart Agriculture

Environmental monitoring goes hand in hand with smart agriculture. Some of the challenges presented in this industry include infrastructure limitations, labor challenges, and economic challenges. Many rural areas also lack reliable internet access for the installation of smart agriculture devices. These devices are similar to the LoRaWAN devices discussed previously, except that they use common internet technologies to transmit data, and this can be quite a challenge to set up. With the deployment of a VISTA+ instance, smart agriculture could be easily brought to an area of up to 22 kilometers with a single gateway. The help videos on the platform would help agricultural managers easily add sensors and see data flowing in, mitigating the challenge of having to hire additional staff to manage the technology. The AI summaries could help them make inferences about the crop season, along with conditions of the soil and the surrounding area. With the wide area capabilities of the LoRaWAN network, multiple agricultural sites could be connected to divide costs even further for individual farms.

6.3.2 Medicine and Healthcare

Remote patient monitoring is also a possibility with the work presented in this thesis. Existing Lora devices for monitoring heart rate, blood pressure, and body temperature could be integrated with the VISTA+ portal. Doctors could then remotely access patient data and, with the help of AI, efficiently summarize daily readings to ensure patient health and safety. For repetitive tasks of checking daily readings, the suggestive actions feature could allow them to do the task with a single click.

6.4 Scenario C: Civic and Educational

6.4.1 Neighborhood Weather Watch

Neighborhood weather watch is a prime example of civil and educational uses of the VISTA+ innovation via community science. Residents in a community could set up a local instance of VISTA+ to monitor neighborhood climate and weather patterns. This could provide the community with a neighborhood weather reporting service that could help with local event planning. Multiple houses could be given access to the portal, allowing anyone to use the AI-analysis feature to interpret otherwise complex data. The new map page, combined with multiple soil moisture sensors, could help them track water runoff in neighborhoods with various slopes and elevations.

6.4.2 Urban Gardening

Urban gardening is the use of citizen science to help with monitoring the growth of plants at a small scale. This scenario could apply to a single person monitoring plant growth at their home or in the educational sector by setting up low-cost sensors at school to monitor the health of a school garden. Both examples would greatly benefit from the new features on the VISTA portal that will make their experience while using the portal much better. From an educational perspective, students could be taught concepts related to environmental science without teachers needing to focus on the details about the data pipeline. Students could ask the LLM questions like "What has changed in the garden's soil over the past week" to get a response that could help them learn about soil composition concepts. The possibilities within education are substantial.

6.4.3 Data Exploration for Students with Disabilities

One of the most important objectives of this thesis was to improve the general and data accessibility of the VISTA portal. With the changes implemented for VISTA+, the VISTA portal is now more accessible to people with disabilities than ever before.

The proposed scenario allows people, and specifically students with visual or learning impairments, to successfully navigate and perform tasks on the application just as efficiently as anybody else. Instead of relying on inaccessible graphs not supported by most screen readers, students can use the AI assistant to textually summarize major trends and perform other statistical explorations.

Chapter 7

User Study and VISTA+ Portal Evaluation

7.1 Objective

This user study focuses on comparing traditional data presentation approaches (e.g., graphs and tables) with the AI-analysis feature on the VISTA+ portal. The goal is to evaluate the usability of the natural language summaries and insights to determine whether or not it can serve as a complementary tool to help users make better data inferences. A secondary objective is to measure whether the AI improved efficiency and data interpretation capabilities and whether those benefits scale with task complexity. The goal is not to evaluate whether the data visualization methods are better than the AI approach or the other way around, as the goal behind integrating an LLM with the data was for the tool to serve as support for traditional data visualization approaches.

This chapter details the user study parameters, including the participants, general setup, procedure, questionnaires, and the tasks given to the participants. The study's within-subject design is also explained in detail. The chapter later goes on to provide results from the user study by performing parametric and non-parametric tests to study the relationship between the independent and dependent variables. The chapter wraps up by discussing the results and, largely, what can be extracted from the user study to better enhance the VISTA+ portal.

For the user study presented in this chapter, approval from the Institutional Re-

view Board (IRB) was acquired under application number 2262166-2 at the University of Nevada, Reno.

7.2 Experimental Setup

7.2.1 Participants

The participants for this user study were students from the University of Nevada, Reno. Twenty-five students participated in the study, with six of the twenty-five participants being undergraduate students. 100% of this pool of participants were studying computer science and engineering at an undergraduate or graduate level. The age groups reported by the participants were 18-24 (14 participants), 25-34 (10 participants), and 35-44 (1 participant).

The participants were chosen with the demographics above to focus the user study towards graduate-level researchers, one of the two intended users. The goal was to achieve a 75% user study participant pool comprising graduate students, and the goal was achieved by recruiting 19 graduate students (76%). The participants were recruited through mutual connections from the researchers. After finding a potential participant, they were sent or given a brief prompt about the user study. Other participants were recruited via various modes of communication (email, text, etc.).

7.2.2 Setup

A modified version of the VISTA+ portal was implemented for the sole purpose of the user study. The only features kept from VISTA+ were the AI-Analysis tools on the "Export Data" and "Data Visualization" pages. This software differed from the VISTA+ portal in being able to take quantitative measurements during the study. Some of those metrics are the number of clicks, the accuracy of tasks, and total execution time. Changes to the front-end for this modified version of the VISTA+ portal were limited to a group of buttons in a small space on the top right corner

of the two pages where the tasks were performed. A uniform data pool was applied to keep the results relatively consistent. The data was pulled from real soil moisture and temperature sensors deployed at one of the developers' houses.

The hardware needed for the study is a computer with the modified VISTA+ software running on a locally hosted server. This computer was set up so that the software was running before the participant arrived. The laptop was connected to a monitor, an external keyboard, and a mouse for the participant to use. The split-screen design helped the researcher take readings from the screen while the participant read about the next task. The same computer in the William N. Pennington Engineering building, room 436, was utilized for all participants to ensure no performance differences from participants using different hardware. For this reason, conducting the user study virtually was not possible. The pre-study and post-study questionnaires were digitally accessed on the same computer. A list of tasks was printed to help the user refer back to the task without having to switch tabs or focus on a different digital page.

7.2.3 Design

The user study was made to be a within-subjects design. This is where participants experience both conditions (data presentation tools and AI analysis). The study was constrained to be twenty-five to thirty minutes long so that recruitment without any monetary benefit (on a volunteer basis) could be possible. During several preliminary test studies, it was determined that six tasks would be possible with this time constraint, given that the participant also had to fill out two questionnaires and be given a short software demo. Six tasks also allow for an even number of tasks to be done with each tool.

Another important design choice to consider is the level of tasks. The six tasks were split into three levels as indicated in Subsection 7.2.5. The levels were introduced to gradually scale the difficulty of tasks, where level one would be to familiarize the participant with the software, and the two subsequent levels would increase in

challenge, therefore increasing the cognitive load of the participant. The reasoning behind this design choice is simple: to find out if the use of artificial intelligence increases as the tasks get more complex and challenging.

The study had two independent variables with two and three factors respectively: presentation tool (Traditional Data Presentation and AI) and Task Level (levels 1, 2, and 3). These were the variables being changed throughout the study. The study had three dependent variables that are automatically computed by the software: accuracy (percent of correct answers, task completion time (in seconds), and input device actions (mouse clicks). The main method of counterbalancing was to have the participant utilize an alternative presentation tool for every succeeding task. There is evidence present in research suggesting that iterating between visualizations isn't ideal for user studies [6], but since the participant was alternating between one of two tools, the risk remained minimal.

7.2.4 Procedure and Questionnaires

The procedure was constructed to ensure consistency among participants, and the following procedures were given to each researcher conducting the user study. Here is a timeline of events after the participant arrives:

1. The participant is given a brief introduction to the user study without disclosing exactly what the tasks entail.

2. The participants were shown a consent form for participating in the user study. They were not required to sign the consent form.

3. The participant is navigated to a Google form for the pre-study questionnaire. The participant is to fill out this form before proceeding.

4. The participant was given a brief demo of the software on the computer being utilized for the user study. More detail was given for the two pages that contain the tabular view of data (Export Data page) and the visualization view of data

(Data Visualization page), showing them how they will be completing the tasks. The AI-analysis tool was also demoed, so the user knew how to use it before the tasks began.

5. The participants were given time to explore and ask any questions before starting the tasks.

6. At this point, the participant begins their tasks. The researcher conducting the user study monitored and answered questions without intervening too much.

7. Once the participant had completed the tasks, they were navigated to the post-study questionnaire.

8. While the participant was completing the second questionnaire, the researcher reviewed the data to make sure all values were properly recorded and not missing.

9. The participant was thanked, dismissed, and a data compilation process started. For consecutive studies, data were processed at the end of the day.

The pre-study questionnaire collected both quantitative and qualitative metrics. The form gathered demographic data, assessed prior usage of AI and data analysis tools, and evaluated the general proficiency of the participant. Here are the questions asked of the user with the method of response:

1. What is your age group? Multiple choice.

2. What is your gender? Multiple choice.

3. What is your academic major or field of study? Short Answer.

4. What year are you in your program? (e.g., undergraduate first year, second year, graduate). Multiple Choice.

5. How would you rate your overall proficiency with computers? Likert scale ranging from 1 to 7.

6. How familiar are you with data analysis tasks? Likert scale ranging from 1 to 7.

7. How familiar are you with using artificial intelligence? Likert scale ranging from 1 to 7.

8. Do you regularly work with data or statistical analysis as part of your coursework or research? Likert scale ranging from 1 to 7.

9. Have you used any specific data visualization software or tools in the past? If so, which ones? Long Answer.

10. Have you used any large language models or tools in the past? If so, which ones? (Chat-GPT, Gemini, DeepSeek, etc.). Long answer.

11. What would you expect an AI assistant to help with when analyzing a dataset? Long answer.

12. Do you typically work on data tasks on a particular type of device? Multiple choice.

13. Do you have any disabilities? You may choose not to answer. Checkboxes with options "I prefer not to say", "Learning", "Vision", "Physical", and "None". This question would later help evaluate the tools for use by people with disabilities.

The post-study questionnaire assisted in getting feedback from the user about how the user study went and how they felt about the features of the VISTA+ portal. This would help assess the user experience after a short period of use. Below are questions asked of the user in the post-study questionnaire, with the method or response for each question:

1. Please rate how comfortable you were during the test. Likert scale ranging from 1 to 7.

2. How confident do you feel that you accurately completed the tasks using the visualization tools (table and graph)? Likert scale ranging from 1 to 7.

3. How confident do you feel that you accurately completed the tasks using the AI-analysis tool? Likert scale ranging from 1 to 7.

4. Please rate your overall user experience during the user study. Likert scale ranging from 1 to 7.

5. Overall, which tool was easier to handle during the tasks? (Data visualizations / AI-Analysis / No preference) And what specific aspects made this tool easier to use? (e.g., less cognitive load, enjoyment, interactions, etc.). Long answer.

6. In a real-world setting outside of this study, which tool would you prefer to use for similar data analysis tasks (Data visualizations / AI-Analysis / No preference) and why? Long answer.

7. What were the main challenges you faced when using the data visualization tools? Long answer.

8. What were the main challenges you faced when using the AI-Analysis tool? Long answer.

9. Do you have any suggestions for improving the data visualization tools to make them more effective for similar tasks? Long answer.

10. Do you have any suggestions for improving the AI-Analysis tool to make it more effective for similar tasks? Long answer.

11. Do you think AI-driven interaction could replace traditional tools for data analysis? Why or why not? Long answer.

12. Any other comments or suggestions? Long answer.

7.2.5 Tasks

The tasks were split into three difficulty levels, with two tasks per level. Level 1 tasks were meant to familiarize the participant with the portal and the two methods of data analysis (data visualization and AI). These were tasks that had participants simply identify a pattern or measurement from the data. Level 2 tasks were a bit more complex, combining measurement identification and a secondary analysis portion. Level 3 tasks took it a step further by comparing two different measurements and asking participants to perform a basic analysis on top of the measurement identification. Here is the full list of tasks printed for the user:

- Level 1 tasks
    1. (Data Table) Select the 'laird-temp' device. Find the highest recorded temperature in the past 7 days.
    2. (AI-Analysis with Table) Select the 'dragino-soil-moisture' device. Find the lowest recorded soil moisture in the past 7 days.
- Level 2 tasks
    1. (Data Graph) Select the 'laird-temp' device with reading 'temperature' for the past 24 hours and answer the following question: What time of day does the temperature tend to be the highest?
    2. (AI-Analysis with Graph) Select the 'dragino-soil-moisture' device with reading 'soil-moisture' for the past 7 days and answer the following question: What was the range (min–max) of soil moisture in the past 7 days?
- Level 3 tasks
    1. (Data Graph) Select the 'dragino-soil-moisture' and 'laird-temp' devices with readings 'soil temperature' and 'temperature' for the past 7 days and answer the following question: Which reading (soil temperature or

temperature) had the most variability (difference between the highest and lowest point)?

2. (AI-Analysis with Graph) Select the 'dragino-soil-moisture' and 'laird-temp' devices with readings 'soil moisture and 'temperature' for the past 7 days and answer the following question: Which reading (soil moisture or temperature) had the most variability (difference between the highest and lowest point)?

7.3 Results

The results from the conducted user study are divided into parametric testing and non-parametric testing.

7.3.1 Tests Conducted

Parametric Tests

Eight one-way Analysis of Variance (ANOVA) tests were conducted using parametric interval data collected regarding each participant's task performance. These were chosen for the user study as the design was within-subjects. Two out of three dependent variables, *task completion time* and *number of input interactions (mouse clicks)*, were the focus of these tests. Accuracy was omitted from the one-way analysis of variance since there was no variance for this measure across the different types of presentation tools. Each of the two dependent variables that were focused on were studied for each level of task (3) and separately for all tasks combined. Each iteration of the test compared *Data Presentation* and *AI*. All ANOVA tests were run with $\alpha = 0.05$ to enforce a strong threshold of statistical significance. For these comparisons, pairwise t-tests with $\alpha = 0.05$ were used. The results of these comparisons form the basis of the objective conclusions described later in the discussion portion of this chapter.

Three two-way ANOVA tests were also run to simultaneously examine the impact

of two independent variables along with their interaction. The focus of the three tests was the three dependent variables. This set of ANOVA tests followed a 2 (interface) x 3 (Task Complexity) factorial design.

Non-Parametric Tests

For the non-parametric testing, three Kruskal-Wallis tests were performed between pre-study and post-study questionnaire data. For any tests showing a statistically significant difference, additional tests were performed to determine which conditions differ from which other test conditions. This test was chosen as the data contains a continuous variable across levels of a grouping variable. For example, task execution time and the 7-point Likert scale for computer proficiency. These tests are also run with $\alpha = 0.05$, and if $p < 0.05$, it can be concluded that there is a statistically significant difference.

7.3.2 Findings

ANOVA

Of the eight one-way repeated measures ANOVA parametric tests conducted, six were statistically significant, and the other two failed to meet the $\alpha = 0.05$ threshold. First, ANOVA tests were run for the *Number of Clicks* dependent variable for every level of task present. This was followed by a test for the number of clicks across all tasks, split by the presentation tool used by each participant. The next set of ANOVA tests was run for the *Task Time* dependent variable for all three levels of task separately. Finally, the last ANOVA one-way test was run for the completion time across all tasks, split by the presentation tool used by each participant.

Table 7.1 represents the results for the *Number of Clicks* by task level. DV is meant to refer to the traditional data visualization approaches (tables and graphs), and AI stands for the artificial intelligence analysis tool on the VISTA+ portal. For level 1 tasks, the mean value of the number of clicks using the data visualization tool was 15.76 clicks, about 10% higher than the 14.20 clicks for the AI analysis tool. All three tests in this table were significant with $F_{(1,24)} = 5.613, p < 0.05$,

Table 7.1: ANOVA results for number of clicks by task level.

Task & Group	Mean	Var	F	p	Sig?
Level 1 - DV	15.76	14.36	5.613	0.0262	Yes
Level 1 - AI	14.20	10.75			
Level 2 - DV	9.76	1.02	27.670	< 0.0001	Yes
Level 2 - AI	11.64	2.82			
Level 3 - DV	12.84	9.47	10.619	0.0033	Yes
Level 3 - AI	15.72	12.21			

Table 7.2: ANOVA results for number of clicks across all tasks.

Group	Mean	Var	F	p	Sig?
Data Presentation	12.79	6.20	6.39	0.0185	Yes
AI	14.25	3.91			

$F_{(1,24)} = 27.670, p < 0.05$ and $F_{(1,24)} = 10.619, p < 0.05$ respectively for level one, two and three tasks. This means there was an impact of the data presentation tool on the number of clicks for all three levels. Table 7.2 demonstrates the result of running a one-way ANOVA test for all tasks to measure the difference in number of clicks impacted by the data presentation tool. This result was also significant $F_{(1,24)} = 6.39, p < 0.05$.

The next set of tests was run similarly to the number of clicks, but instead for the task completion time dependent variable. The results for this are shown in Table 7.3. The mean completion time for level 1 tasks using the data visualization tool was 12.20 seconds, much lower than the mean completion time for level 1 tasks with the AI analysis tool. This was statistically significant ($F_{(1,24)} = 34.665, p < 0.05$). Level 2 tasks compared for completion time showed a similar pattern, with the variance decreasing when analyzed against level 1 tasks. This was also a significant difference ($F_{(1,24)} = 14.014, p < 0.05$). The difference between level 3 tasks for the data presentation tool was insignificant, but the mean task completion time for the artificial intelligence analysis tool (43.30 seconds) was lower than the data visualization mean task completion time (60.05 seconds). The last one-way repeated measures ANOVA test was run to measure the difference in task completion time for all tasks overall.

Table 7.3: ANOVA results for task completion time by task level.

Task & Group	Mean	Var	F	p	Sig?
Level 1 - DV Level 1 - AI	12.20 32.25	30.56 294.09	34.665	< 0.0001	Yes
Level 2 - DV Level 2 - AI	19.35 28.25	164.43 164.28	14.014	0.0010	Yes
Level 3 - DV Level 3 - AI	65.21 47.79	4009.03 688.89	1.900	0.1808	No

Table 7.4: ANOVA results for task completion time across all tasks.

Group	Mean	Var	F	p	Sig?
Data Presentation AI	32.54 36.87	621.30 182.46	0.65	0.4276	No

The result was insignificant, meaning that the choice of presentation tool had no impact on the completion time (Table 7.4).

The results of the two-way ANOVA tests are summarized in Table 7.5. A total of four relationships were found to be significant.

1. **Task Completion Time (F1)**: This indicates that interface type (data visualization and AI analysis) had a strong effect on how long tasks took.

2. **Task Completion Time (F2)**: This indicates that task complexity (levels 1, 2, and 3) had a strong effect on how long tasks took.

3. **Number of Clicks (F1 X F2)**: This relation ship was also found to be significant ($F_{(2,48)} = 54.033, p < 0.05$). This means that the interface and task complexity jointly influenced the number of clicks made by the participant.

4. **Accuracy (F1)**: One-way ANOVA tests were not run for the *Accuracy* dependent variable as the accuracy between the tools was quite identical. A two-way test also hints this as the test establishes a weak but statistically significant difference. This indicates that the interface (data visualization and AI analysis) had some effect on how accurately the participant answered.

Kruskal-Wallis Tests

Table 7.5: Summary of two-way repeated measures ANOVA results with significance for task completion time, number of clicks, and accuracy.

Measure	Effect	F(df)	F	p	Sig?
Task Completion Time	Interface (F1)	F(1, 24)	30.377	$< .0001$	Yes
	Level (F2)	F(2, 48)	16.371	$< .0001$	Yes
	Interface × Level (F1 × F2)	F(2, 48)	1.169	$> .05$	No
Number of Clicks	Interface (F1)	F(1, 24)	0.156	.6960	No
	Level (F2)	F(2, 48)	2.040	.1411	No
	Interface × Level (F1 × F2)	F(2, 48)	54.033	$< .0001$	Yes
Accuracy	Interface (F1)	F(1, 24)	4.571	.0429	Yes
	Level (F2)	F(2, 48)	1.000	.3754	No
	Interface × Level (F1 × F2)	F(2, 48)	1.000	$> .05$	No

Pre-Study Questionnaire Test This test was based on a question asking the participant to rate their level of computer proficiency. The participants could answer this question via a 7-point Likert scale, and the frequency of collected data across all participants is shown in Figure 7.1. A Kruskal-Wallis test was performed to study the relationship between reported computer proficiency and average execution time observed for the participant. This test turned out to be statistically significant ($p < 0.05$). At this point, the only takeaway from this is that participants' execution time differs significantly based on their reported computer proficiency. To further analyze this phenomenon, three pairwise Mann-Whitney U tests were performed for all combinations of proficiency that were reported (five, six, and seven). The results from those tests are mentioned in Table 7.6. The only significant result ($p < 0.05$) was the comparison between reported values of Proficiency six and seven. This is an indicator that those participants who reported a proficiency of seven likely completed the tasks faster than those who reported a proficiency of six.

Post-Study Questionnaire Tests Two more Kruskal-Wallis tests were performed on data extracted from the post-study questionnaire. Two of the questions from the questionnaire asked the participant about their confidence using either the data visualization (graphs and tables) or the AI analysis feature. These two metrics,

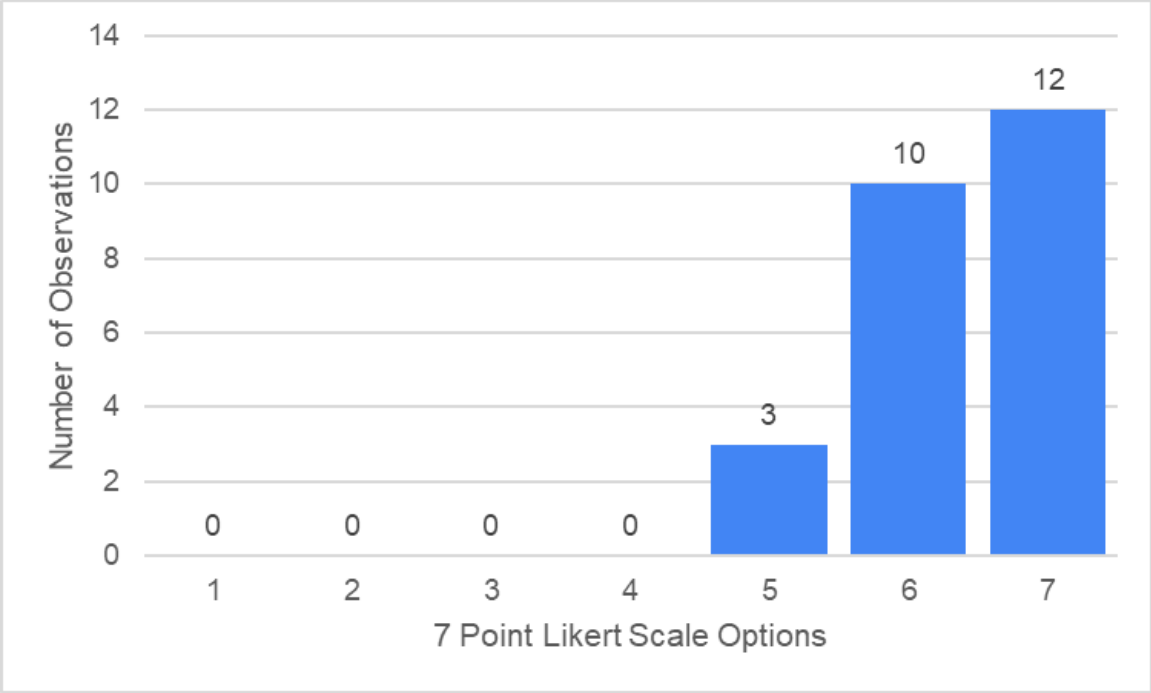

Figure 7.1: Bar chart of computer proficiency ratings from pre-study questionnaire.

Table 7.6: Mann-Whitney U test results comparing execution time across proficiency levels.

Group Comparison	**U Statistic**	**Z Score**	**p-value**	**Significance**
Proficiency 5 vs 6	9.0	-1.014	0.3105	Not Significant
Proficiency 5 vs 7	18.0	0.000	1.0000	Not Significant
Proficiency 6 vs 7	15.0	-2.967	0.0040	Significant

which were reported on a 7-point Likert scale, were compared to their respective task completion times associated with using each interface. For example, the average time each participant took to complete the tasks using AI was compared to their reported confidence level of using AI to answer the task. The results from both tests were insignificant ($p = 0.880$ for data visualization and $p = 0.311$ for the AI tool. This means that there was no strong evidence of a difference in completion time between participants grouped by their confidence levels in using AI or the data visualization tool.

Other interesting results:

1. 24 out of 25 participants mentioned Chat-GPT as one of the AI tools they have used previously.

7.3.3 Participant Comments

As described in Subsection 7.2.4, the questionnaires asked questions to which the participant responded with short and long answers. Here are some of the more salient responses to those open-ended questions from both the pre-study and post-study questionnaires.

Q: What would you expect an AI assistant to help with when analyzing a dataset?

"I would expect an AI assistant to show me the best overall graphs for analyzing a specific dataset (e.g. if time series data then a line chart). I would also expect an AI to give the dataset's "story" or the main message that can be received from viewing a visualization or the dataset as a whole."

"To give me a high-level overview about the structure of the data and any obvious patterns that might be of interest."

"I personally do not expect a AI assistant to help in analyzing a dataset since I see data analysis as an activity not aptly tuned for a AI assistant. However, I do see it helpful when attempting to find or finding a way to create a model for a dataset (e.g., statical model)."

Q: Overall, which tool was easier to handle during the tasks? (Data visualizations / AI-Analysis / No preference) and what specific aspects made this tool easier to use? (e.g., less cognitive load, enjoyment, interactions, etc.)

"Overall, the traditional approach was easier than the AI-assisted approach. Some tasks are simply better suited to the traditional approach, particularly when you are quickly absorbing information from visualized data. This is just quicker than reading a dense paragraph (or attempting to formulate a proper prompt to get a concise explanation). In scenarios where the data was difficult to immediately ingest

on my own (like a crowded graph), the AI was helpful when it properly interpreted my prompts."

"Overall, I believe I had an easier experience handling the tasks using the ai-analysis. Being able to instantly retrieve the information by asking with a prompt or viewing the AI-generated analysis without having to explore the visualization or table saved me some time and was an easier task than navigating the tables and graphs."

Q: In a real-world setting outside of this study, which tool would you prefer to use for similar data analysis tasks (Data visualizations / AI-Analysis / No preference) and why?

"It depends. I feel like if I were to use data visualizations, I would understand what I am actually looking at and make my own determination based off of what I see, however, this could introduce error. The AI-analysis is easier to use and would not introduce the same error as if I looked at a visualization, but I am not actually looking at the data which makes me feel... icky for lack of a better word. I do not have inherent trust in AI tools so I think a combination of both would be preferred in a real-world setting."

Q: Do you think AI-driven interaction could replace traditional tools for data analysis? Why or why not?

"I think it definitely could, but there are issues of trust in AI as well as the fact that analyzers would not be looking at their data to cross-check the AI. I think a combination of both tools is the best solution for now, until there is a complete trust and research that there are LESS or no mistakes at all when analyzing data with AI."

"I think AI-driven interaction could absolutely supplement traditional tools, but I do not think it should replace them. There is some inherent distrust in AI systems because they have the capacity to hallucinate or misinterpret user instructions. They can provide a great feature of convenience for some users, but the option to use traditional tools is necessary for transparent systems."

A preliminary look at the data revealed an irksome UI complication. Several participants called it out in the post-study questionnaire, saying "The only challenge

was not entering the enter button" or "pressing enter was not an option". The chat interface would close if the user pressed the Enter key for chat submission instead of the send chat button. A minor issue like that was very bothersome for some of the participants, and it was fixed after the fifth user study session.

7.4 Discussion

The user study evaluation of the VISTA+ portal's AI-analysis feature revealed insightful findings that report the current quality of the portal and feature. The study also collected feedback that will help with the portal's further development. Now that the results have been reported, what can be said about the quality and innovation of using AI for not only the VISTA+ portal but also other sensor-based platforms?

Starting with the very most basic reported result. It was found that out of the 150 tasks run amongst all the participants (6 each), there were only four instances where any participant answered the task wrong; the accuracy had no variance whatsoever, as the wrongful answers were equally divided between AI and data visualization. While this metric could easily change with more participants, it still indicates that the AI analysis tool is usable enough to gain similar data insight as other traditional data visualization approaches. Parametric tests take this one step further and suggest that AI might be even more beneficial as the complexity of tasks increases. This is evident by the shifting pattern in the mean task completion time difference between AI and data visualization. For level 1 and level 2 tasks, the ANOVA variance was significant, and looking at the mean completion time revealed that data visualization tools (graphs and tables) took less time than AI. With level 3 tasks, however, the result is no longer significant, and the data suggests that the mean completion time for tasks with AI started to take less time than data visualization. While insignificant, future studies with more complex tasks might help completely test this dynamic pattern.

Other parametric tests were conducted to study the dependent variable *Number of Clicks*. The number of clicks is a measure of efficiency, cognitive load, and general usability. Overall, the data visualization tool tasks took less number of clicks to

complete the task. This indicates room for improvement within the UI design for the analysis feature. It is important to note that the AI analysis feature also mandates the use of the keyboard to type into an input. This is another means of interaction with the software, but it was not tracked during the study.

The final set of parametric tests was conducted via two-way Analysis of Variance (ANOVA) examinations. These tests helped confirm the other one-way ANOVA tests, revealed a surprising metric about accuracy, and helped to see the interaction effect of the number of clicks. Accuracy was not tested with one-way ANOVA tests, as there was no observable variance; however, the two-way ANOVA test revealed a weak but significant effect of the interface type on accuracy. This was because of how variance is distributed across participants, not just the overall mean. So even though the overall mean had no variance whatsoever, a repeated measures two-way ANOVA test was able to extract that the independent variable of data presentation tool affects overall accuracy. The last big takeaway from this last set of tests further analyzed the number of clicks and found that both the task complexity and interface type jointly impacted the number of clicks the user had to perform. The one-way ANOVA test had already revealed one part of that relationship. Logically, taste complexity impacting the number of clicks makes sense since it can be assumed that with a higher difficulty of the task, more interaction with the software is required to get to the final answer.

Non-parametric tests gave insight into user behavior and background, impacting their performance during the user study. This was very evident with the Kruskal-Wallis test studying computer proficiency and task execution time. It was revealed in the results that if the user reported a score of 7 versus a score of 6 on the Likert scale for computer proficiency, they were more likely to complete the tasks faster. These tests also revealed that participants' confidence in their answers using each tool did not align with their time-based performance for the tasks.

The participants' comments from the questionnaires are arguably the most important for improving the portal. The idea of asking the user beforehand about what

they would expect from an AI tool was a direct suggestion from [3]. This question got a diverse pool of answers, with participants answering anywhere from "it won't work very well", to "To give me a high-level overview about the structure of the data and any obvious patterns that might be of interest". It was interesting to observe the variety of answers and to realize that most of the anticipated UX for the AI analysis feature was covered in the real implementation. The question in the post-study questionnaire, *which tool was easier to handle* and *what tool they would prefer to use outside the study* received similar answers. The responses seemed to be either extremely positive towards one tool or some variation of using both tools. This generally supports the objective of the user study to see if natural language summaries and analysis can serve as a complementary tool for better data interpretation.

Other than that, all the other questions and responses were related to direct feedback on improving the portal and features to better enhance the user experience. The participants guided attention to the weak user interface behind the data graphs, saying that navigating the UI for that was a challenge. The large language model's performance was a common topic for challenges faced when using the AI analysis feature. There were also indicators that the prompts for the LLM were still not optimal, as some participants described the initial response as too long, with the need for a simpler initial response. Other small but valuable user interface nuances and suggestions were pointed out as well.

Overall, the user study provided significant feedback for the user interface and features presented in this thesis. Some of the feedback has already been implemented into the portal, while bigger changes remain future work items for the VISTA+ portal.

Chapter 8

Conclusions and Future Work

8.1 Summary of Contributions

This thesis made three primary contributions that are highly relevant in a world with ever-growing data. The first is the tying of artificial intelligence with real sensor data and using it for data interpretation and insight capabilities. Results from this method have an impact on environmental monitoring, but also on other domains on a large scale. Data accessibility for disabled individuals is another benefit of the first contribution. The second contribution is the sum of artificial intelligence in a wide array of functionalities. This scratches the surface of what's possible in terms of user experience enhancements using AI. The third primary contribution is the introduction of a geographical method of data visualization for sensor-based data pools. The VISTA+ application provides a user-centered interface for effective ecological monitoring. The software serves as a functional presentation of significant research and development outcomes. The result is a more intuitive application that researchers can use to better understand issues and carry out assessments related to environmental monitoring.

8.2 Future Work

Performance was a broad challenge throughout the software, especially with any functionality that involved artificial intelligence. Ideally, to provide the best user experience possible, content using AI would have to be shown at the same time as the

traditional data visualization approach. This means that after the user has selected parameters to fetch their data on either the tabular or graph form of data visualization, the AI response summarizing their data should show just as fast as the data visualization is available. Observations from using AI with a large number of tokens in the prompt indicate that the response time per generated token seems slightly inadequate for AI to serve as a real-time system. While theoretically, some architectures could help with this, such as precomputed responses/caching, model distillation, or progressive enhancements, the simplicity of integration with an application seems to be sacrificed. Implementing one of these solutions to combat the problem of waiting upwards of 15 seconds for an initial AI response is crucial future work.

Other performance challenges include a lagging user interface and occasional mobile device-specific performance issues, so performance is a prime candidate for future work. A single project containing two devices can generate upwards of ten thousand rows of individual data points within a single week, making it difficult for some parts of the front end to contain that much data. This is fixed within the application by introducing data aggregation features for the data graph specifically, but at the sacrifice of true recordings. Mobile device-specific performance issues were observed using virtualization of mobile devices within browser-based development tools. Now and then, after fetching data onto a data visualization tool (before the AI response is ready), mobile devices and devices with processing constraints would tend to delay user touch input on the platform.

Other future work revolves around migrating the entire tech stack to a cloud computing solution. This will help with scaling in the future and will make network management easier. Process for data clean up (deleting unused CSV files or removing very old user activity data) is also important in the list of future works.

References

[1] Shadi Abou-Zahra, Judy Brewer, and Michael Cooper. Artificial Intelligence (AI) for Web Accessibility: Is Conformance Evaluation a Way Forward? In *Proceedings of the 15th International Web for All Conference*, W4A '18, New York, NY, USA. Association for Computing Machinery, 2018. ISBN: 9781450356510. DOI: 10.1145/3192714.3192834. URL: https://doi.org/10.1145/3192714.3192834.

[2] Vladimir Agafonkin and contributors. Leaflet: JavaScript Library for Interactive Maps. https://leafletjs.com, 2025. (Accessed: 2025-04-15).

[3] Allam Hassan Allam and Halina Mohamed Dahlan. User experience: challenges and opportunities. *Journal of information systems research and innovation*, 3(1):28–36, 2013. URL: https://seminar.utmspace.edu.my/jisri/Volume3.html.

[4] Amazon Web Services. Amazon Web Services (AWS). https://aws.amazon.com, 2025. (Accessed: 2025-04-15).

[5] Bjorn Andersson, Dionisio de Niz, William Vance, John Ross, Mark Wotell, Willie Fitzpatrick, and Tuan Bui. What is Determinism? Definitions and Implications for Airworthiness and Critical Software. In *2024 AIAA DATC/IEEE 43rd Digital Avionics Systems Conference (DASC)*, pages 1–10, 2024. DOI: 10.1109/DASC62030.2024.10748739.

[6] Somnath Arjun. Personalizing data visualization and interaction. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*, UMAP '18, 199–202, New York, NY, USA. Association for Computing Machinery, 2018. ISBN: 9781450357845. DOI: 10.1145/3213586.3213590. URL: https://doi.org/10.1145/3213586.3213590.

[7] Chase Carthen, Zach Estreito, Vinh Le, Jehren Boehm, Scotty Strachan, Alireza Tavakkoli, Frederick C. Harris, and Sergiu M. Dascalu. Initial design and implementation of an edge-to-edge lorawan data collection system. In *ITNG 2024: 21st International Conference on Information Technology-New Generations*, pages 241–247. Springer Nature Switzerland, March 2024. ISBN: 978-3-031-56598-4. DOI: 10.1007/978-3-031-56599-1_32.

[8] ChatCSV. ChatCSV: Conversational Analysis of CSV Files Using AI. https://www.chatcsv.com, 2025. (Visitied: 2025-04-15).

[9] Google DeepMind. Gemini: A Family of Multimodal AI Models. https://deepmind.google/technologies/gemini, 2025. (Accessed: 2025-04-15).

[10] DeepSeek. DeepSeek: Open-Source Foundation Models. https://www.deepseek.com, 2025. (Accessed: 2025-04-15).

[11] Alexa K. Jay, Allison R. Crimmins, Christopher W. Avery, Travis A. Dahl, Rebecca S. Dodder, Benjamin D. Hamlington, Allyza Lustig, Kate Marvel, Pablo A. Méndez-Lazaro, Mark S. Osler, Adam Terando, Emily S. Weeks, and Ariela Zycherman. Overview: Understanding risks, impacts, and responses. In A.R. Crimmins, C.W. Avery, D.R. Easterling, K.E. Kunkel, B.C. Stewart, and T.K. Maycock, editors, *Fifth National Climate Assessment*, chapter 1, Washington, DC, USA. U.S. Global Change Research Program, 2023. DOI: 10.7930/NCA5.2023.CH1.

[12] Yongkang Li, Yanying Lin, Yang Wang, Kejiang Ye, and Chengzhong Xu. Serverless computing: state-of-the-art, challenges and opportunities. *IEEE Transactions on Services Computing*, 16(2):1522–1539, 2023. DOI: 10.1109/TSC.2022.3166553.

[13] Kristina McElheran, J. Frank Li, Erik Brynjolfsson, Zachary Kroff, Emin Dinlersoz, Lucia Foster, and Nikolas Zolas. AI adoption in America: Who, what, and where. *Journal of Economics & Management Strategy*, 33(2):375–415, 2024. DOI: https://doi.org/10.1111/jems.12576. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/jems.12576. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/jems.12576.

[14] OpenAI. ChatGPT: An AI language model by OpenAI. https://chat.openai.com, 2025. (Accessed: 2025-04-15).

[15] Suresh K. Peddoju and Himanshu Upadhyay. Evaluation of IoT Data Visualization Tools and Techniques. In *Data Visualization: Trends and Challenges Toward Multidisciplinary Perception*, pages 115–139, Singapore. Springer Singapore, 2020. ISBN: 978-981-15-2282-6. DOI: 10.1007/978-981-15-2282-6_7. URL: https://doi.org/10.1007/978-981-15-2282-6_7.

[16] Radhika Pillarisetty and Pratika Mishra. A Review of AI (Artificial Intelligence) Tools and Customer Experience in Online Fashion Retail. *International Journal of E-Business Research*, 18:1–12, January 2022. DOI: 10.4018/IJEBR.294111.

[17] Michael J. O. Pocock, John C. Tweddle, Joanna Savage, Lucy D. Robinson, and Helen E. Roy. The diversity and evolution of ecological and environmental citizen science. *PLOS ONE*, 12(4):1–17, April 2017. DOI: 10.1371/journal.pone.0172579. URL: https://doi.org/10.1371/journal.pone.0172579.

[18] Antonis Protopsaltis, Panagiotis Sarigiannidis, Dimitrios Margounakis, and Anastasios Lytos. Data visualization in internet of things: tools, methodologies, and challenges. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, ARES '20, New York, NY, USA. Association for

Computing Machinery, 2020. ISBN: 9781450388337. DOI: 10.1145/3407023.3409228. URL: https://doi.org/10.1145/3407023.3409228.

[19] Straits Research. Environmental monitoring market size, trends, and growth forecast 2033, 2024. URL: https://straitsresearch.com/report/environmental-monitoring-market. (Accessed: 2025-04-13).

[20] Elizabeth Salinas, Rony Cueva, and Freddy Paz. A systematic review of user-centered design techniques. In Aaron Marcus and Elizabeth Rosenzweig, editors, *Design, User Experience, and Usability. Interaction Design*, pages 253–267, Cham. Springer International Publishing, 2020. ISBN: 978-3-030-49713-2. DOI: 10.1007/978-3-030-49713-2_18.

[21] Marian Stoica, Marinela Mircea, and Bogdan Ghilic-Micu. Software development: agile vs. traditional. *Informatica Economica*, 17:64–76, December 2013. DOI: 10.12948/issn14531305/17.4.2013.06.

[22] Maria Virvou. Artificial intelligence and user experience in reciprocity: contributions and state of the art. *Intelligent Decision Technologies*, 17(1):73–125, 2023. DOI: 10.3233/IDT-230092. eprint: https://journals.sagepub.com/doi/pdf/10.3233/IDT-230092. URL: https://journals.sagepub.com/doi/abs/10.3233/IDT-230092.

[23] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. In *Proceedings of the 30th Conference on Pattern Languages of Programs*, PLoP '23, USA. The Hillside Group, 2023. ISBN: 9781941652190. DOI: 10.5555/3721041.3721046.

[24] World wide web consortium. URL: https://www.w3.org/. (Accessed: 2025-03-25).

[25] Keke Wu, Emma Petersen, Tahmina Ahmad, David Burlinson, Shea Tanis, and Danielle Albers Szafir. Understanding data accessibility for people with intellectual and developmental disabilities. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA. Association for Computing Machinery, 2021. ISBN: 9781450380966. DOI: 10.1145/3411764.3445743. URL: https://doi.org/10.1145/3411764.3445743.