Project VISTA:
Progress Report Presentation

University of Puerto Rico, Mayagüez

Electrical and Computer Engineering Department

ICOM 5047

September 18th, 2025

Jason Gutiérrez - PM, Diego Green, Ricardo Blanco, Alexander González

Honeywell

JARD TECH

Just A Rad Dev

Agenda

Introduction

- Problem Description
- Updates

Body

- Design Alternatives and Choices
- System Architecture and User Interface
- Project Status and Delays
- Deliverables and Milestones
- Budget Status

Conclusion

Problem Description

- **Disconnected systems**:
  - No stable centralized platform to manage project information and monitor laboratory enviromental conditions.
- **Multiple tools required**:
  - Managers and engineers must switch between platforms to access tasks, project information, and laboratory enviromental conditions.
- **Consequences**:
  - Instability on information transmission, time inefficiency, limited project visibility, and increased monitoring risks (temperature/humidity).

Updates

- **Visitor View**:
  - Replaces Visitor role; simplified as the system Home Page.
- **Security**:
  - Implementing Single Sign-On (SSO) with Active Directory, per Honeywell guidelines.
- **Temperature & Humidity Graphs**:
  - Weekly summaries (avg/high/low) to bar charts and rolling averages, as requested by the client.

Design Alternatives and Choices

- Front-end choice: **Razor Pages + HTMX**

Option	Pros	Cons	Fit for VISTA
React / Angular (SPA)	Rich “desktop-app” UX, big ecosystem	Complex Microsoft SSO integration.	Overkill for read-mostly dashboards
Blazor Server	C# end-to-end, SPA-like feel, easy RBAC	Heavier server load, learning curve, scale limited	Possible, but extra tech risk
Razor Pages + HTMX	Great for Microsoft SSO, on-prem friendly (no CDNs)	Less SPA-like, interactivity added via HTMX/JS	Best balance for VISTA’s use case

Table 1: Front-End Design Alternative Comparison

Design Alternatives and Choices

- Back-end choice: **Python**

Option	Pros	Cons	Fit for VISTA
Python with FastAPI Framework	Rapid development and prototyping.	Slower performance for heavy CPU-intensive tasks.	Best for Jira integration and REST API development.
.NET	Entity Framework simplifying SQL database interactions.	Can create performance bottlenecks during CPU-heavy tasks	Integrated within the Microsoft ecosystem
Node.js	A variety of ORM choices for SQL databases.	Can involve more boilerplate code and a heavier framework	Highly performant for I/O and real-time operations

Table 2: Back-End Design Alternative Comparison

Design Alternatives and Choices

- Database choice: **MySQL**

Option	Pros	Cons	Fit for VISTA
MySQL	Open-source, common	Scaling and heavy load performance require tuning	High
MongoDB	Flexible JSON schema; fast to iterate.	Not SQL/relational; weaker AD/SSO; more ops overhead.	Low
Microsoft SQL	Best with Windows, SSO and tooling.	License cost for mass production; heavier.	Medium

Table 3: Database Design Alternative Comparison

System Architecture

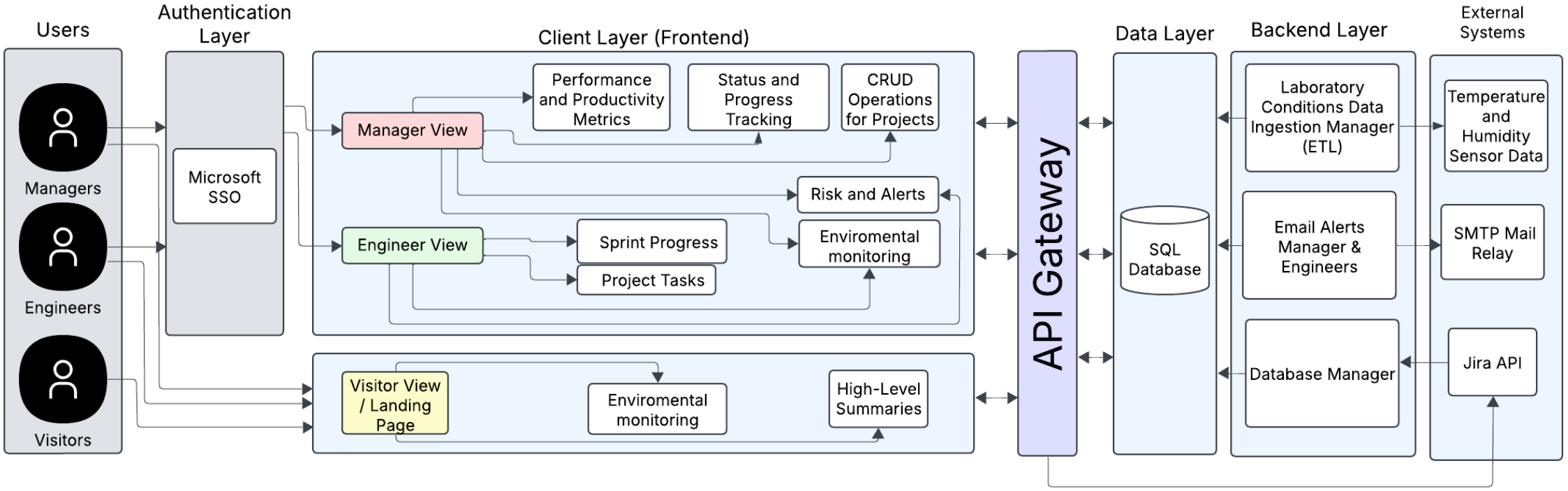


Figure 1: System Architecture Diagram

Front-End Mockups (Login page and Engineer Dashboard)

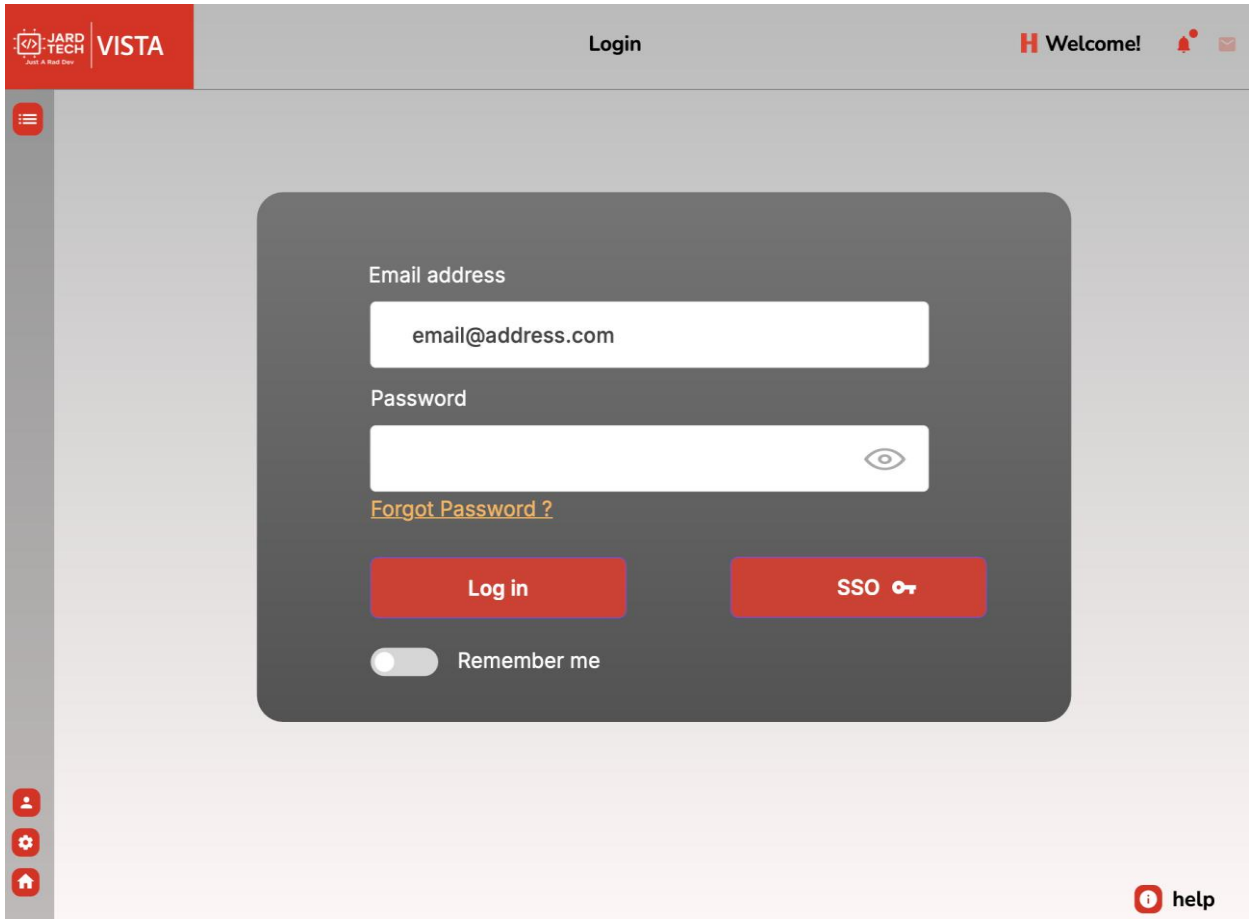


Figure 2: Dashboard Login/SSO Page

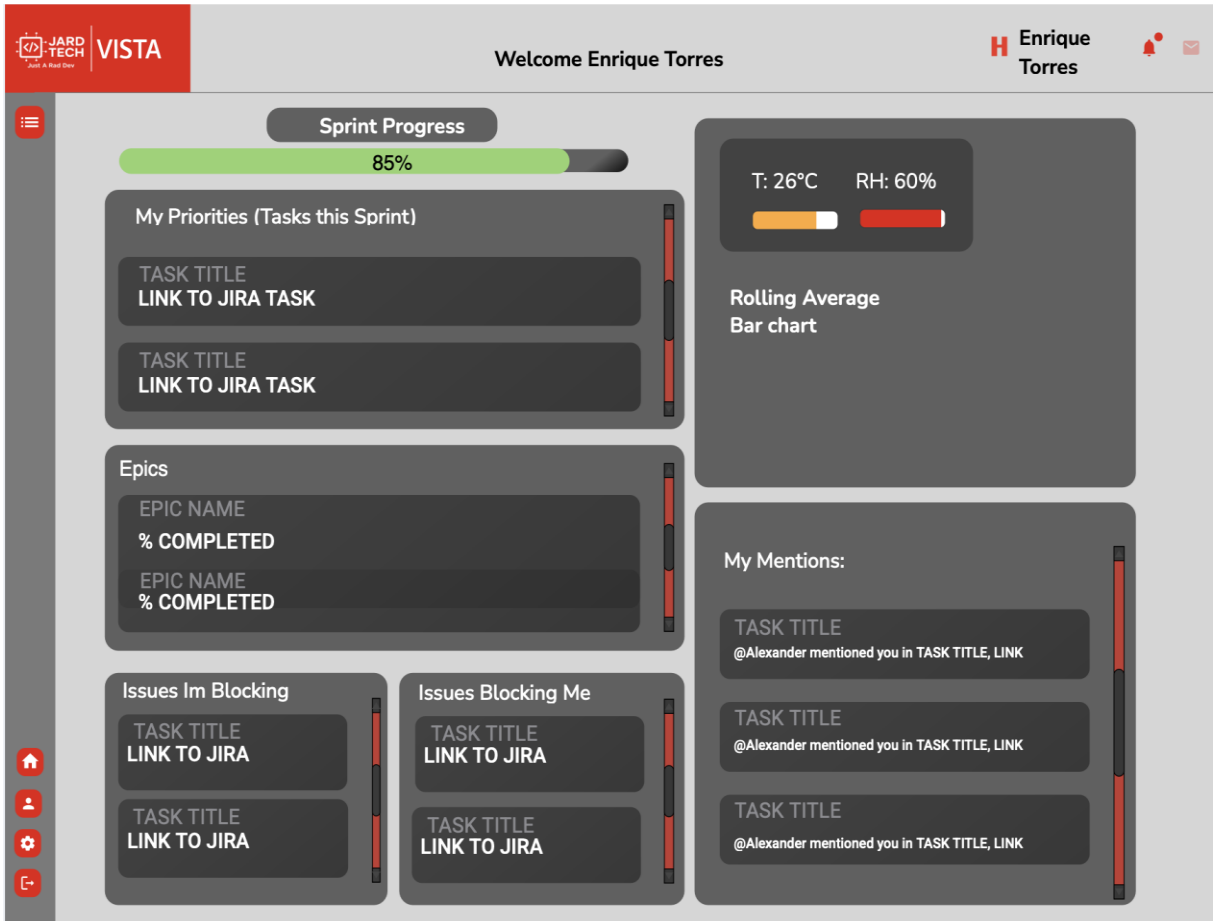


Figure 3: Engineer Dashboard

Front-End Mockups (Manager Overall and Specific Overview)

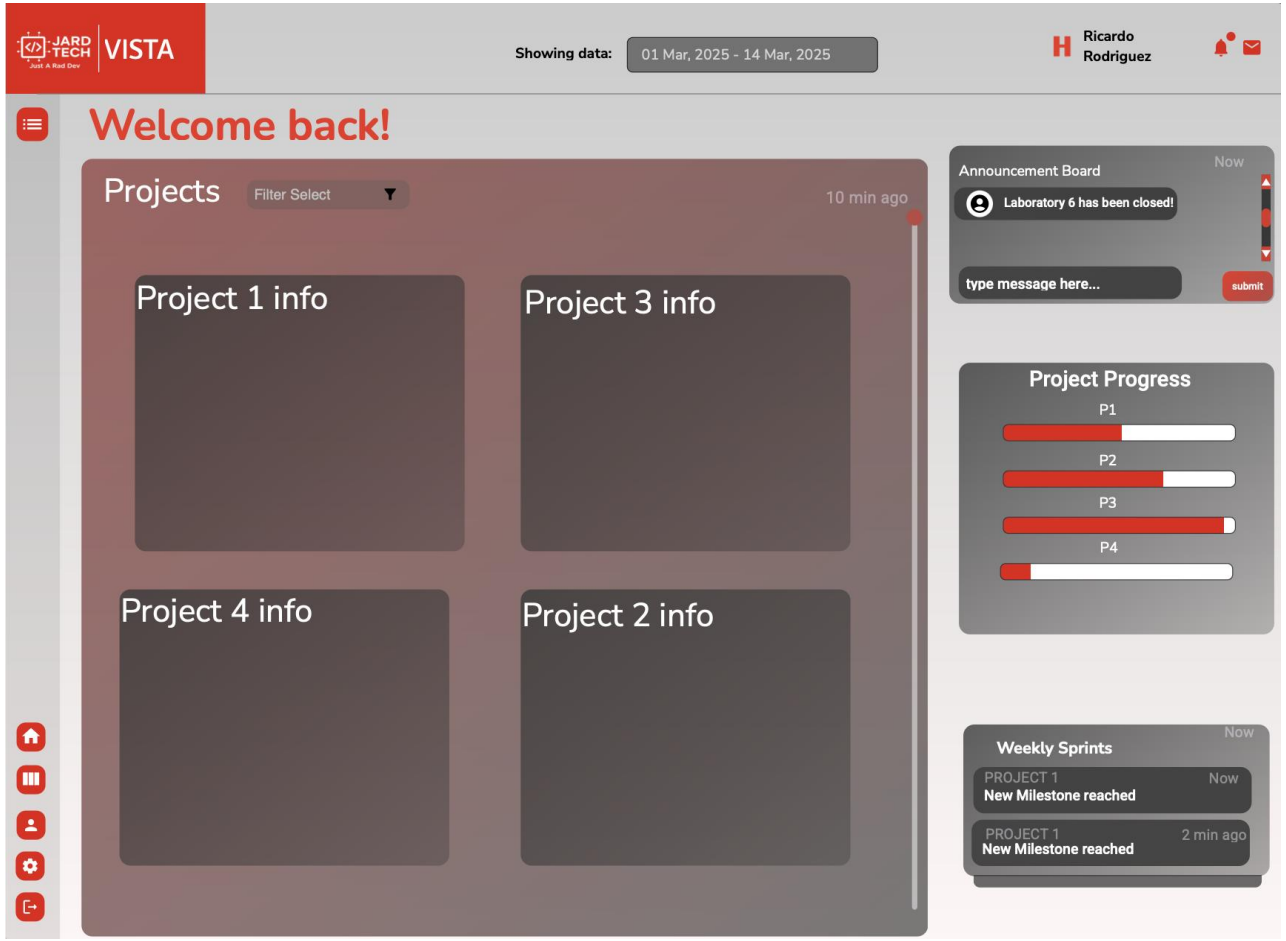


Figure 4: Manager Overall View Dashboard

Figure 5: Manager Specific Overview Dashboard

Project Status

Completed Work:
- System Architecture
- User Interface Design
- Entity Relation & Table Diagram
- Sequence Diagram

In Progress:
- Use Case Diagram

To Do:
- Class Diagram
- Visitor View

Project Delays

Context

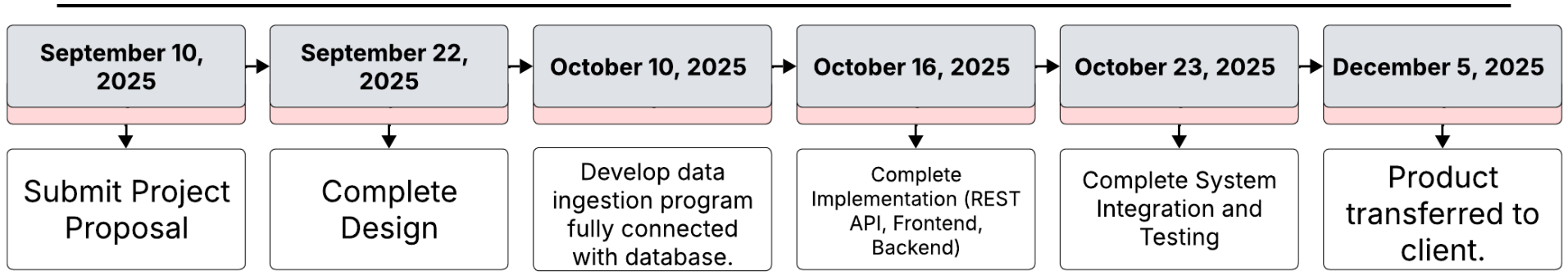
- The third sprint was delayed due to technical and functional questions that needed clarification with the client.

Contingency Actions

- Tasks from the third sprint were moved to the next sprint to avoid blockers.
- Continued working on other designs.
- Technical questions were resolved, and the expected system behavior was clarified.
- Tasks were replanned in the backlog.

Deliverables and Milestones

- **Centralized Web Application:**
  - Provides a stable access point for managers, engineers, and visitors with custom views.
- **Data Management System:**
  - Maintains a database with historical laboratory and project data for analysis and reporting.
- **Project & Laboratory Dashboards:**
  - Displays project summaries (tasks, progress, budget, client, manager in charge).
  - Monitors laboratory temperature/humidity with alerting features.



Budget Status

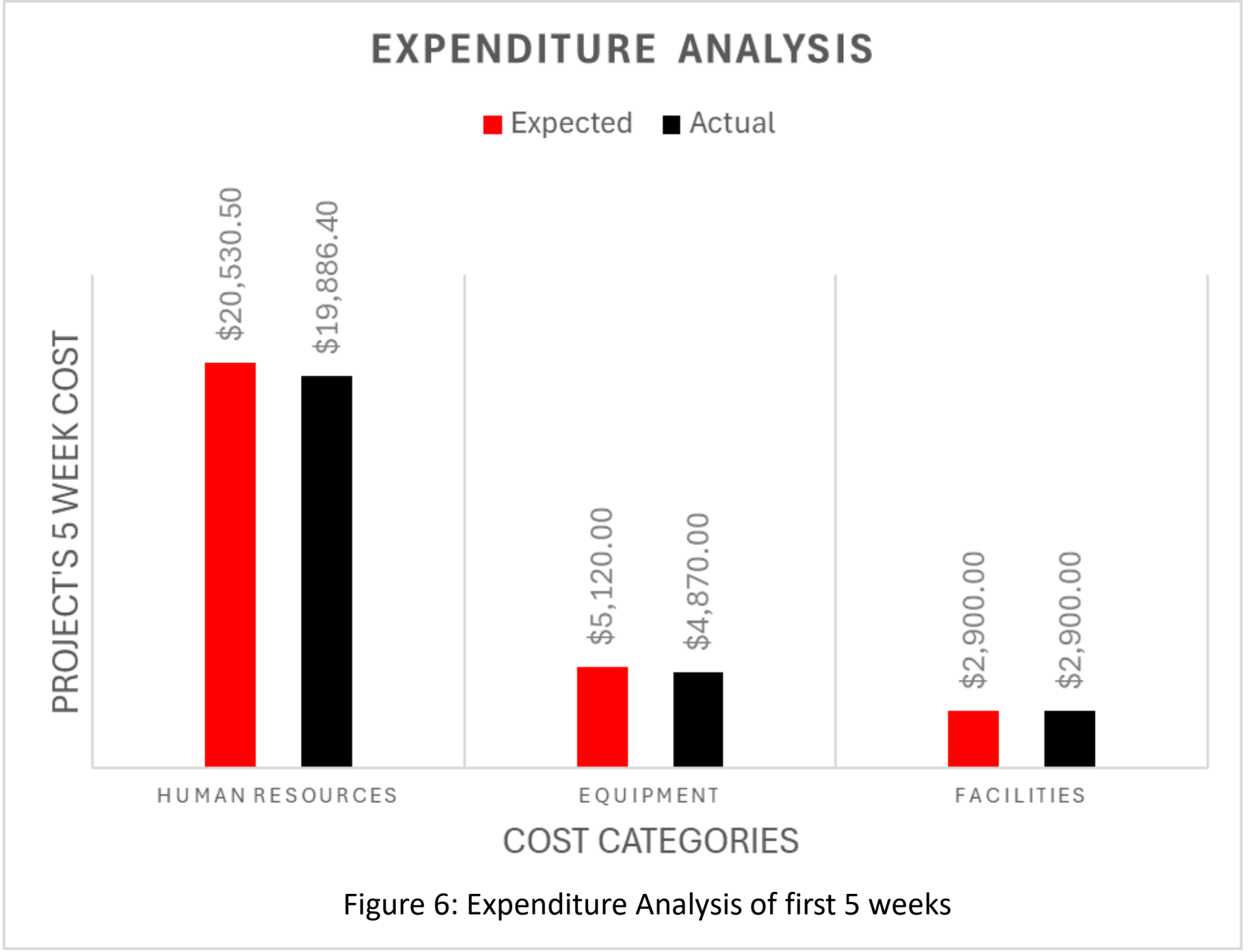


Figure 6: Expenditure Analysis of first 5 weeks

Conclusion

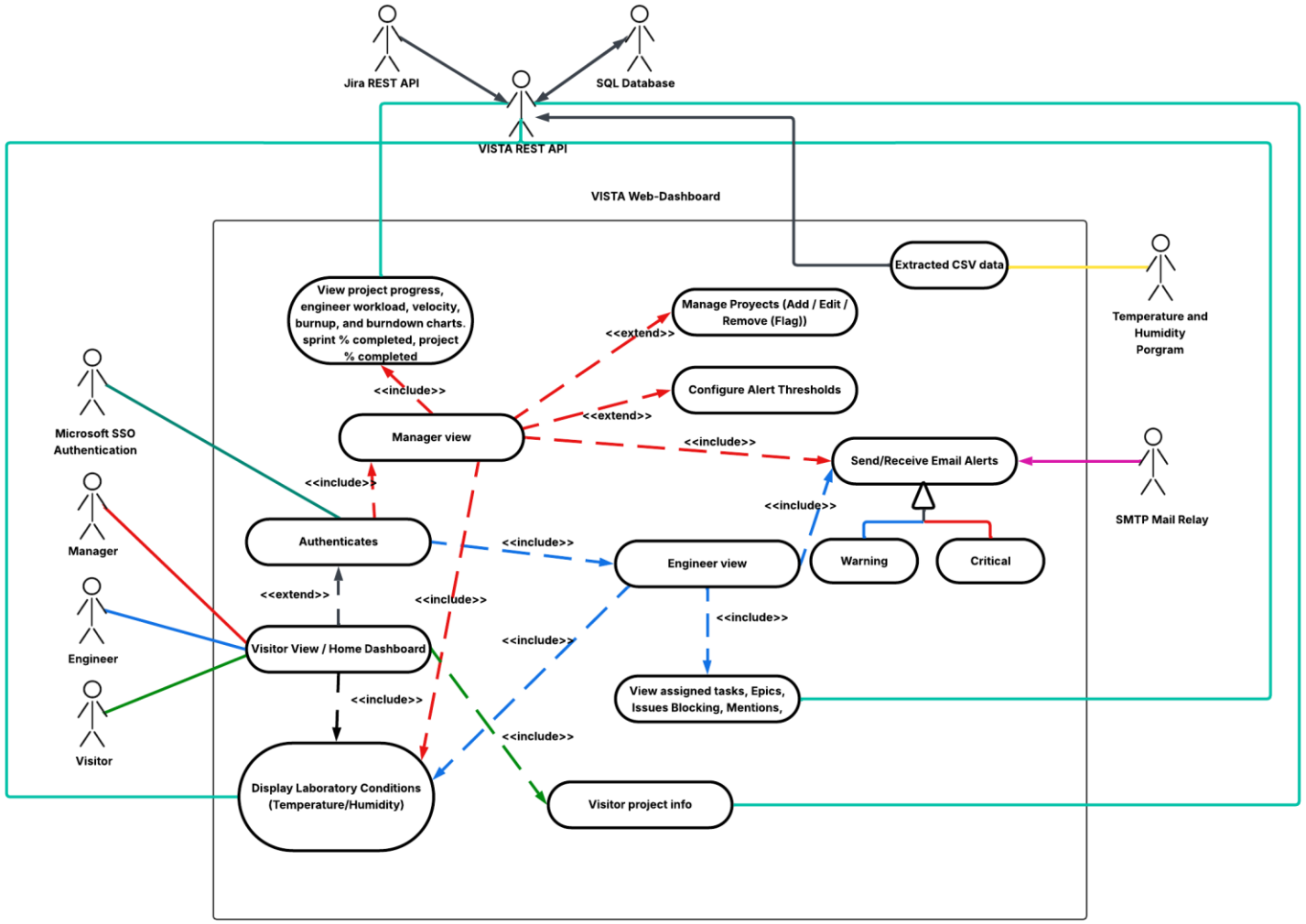
- **Where we are (today)?**
  - Design Phase Almost Done
  - Under budget by 3.13%
- **What are we going to do next?**
  - Finish design
  - Setup dev environment
  - Implement and Configure Database
  - Implement data ingestion and ETL Pipeline
- **Which milestones are coming up soon?**

  Sept 22 - Finish design.

  Oct 10 - Develop data ingestion program fully connected with database.

QUESTIONS?

UML Use Case Diagram



Appendix A: Interaction of Users with Vista Web-App and automatic data synchronization.