Project Management Application with Zero Trust Security

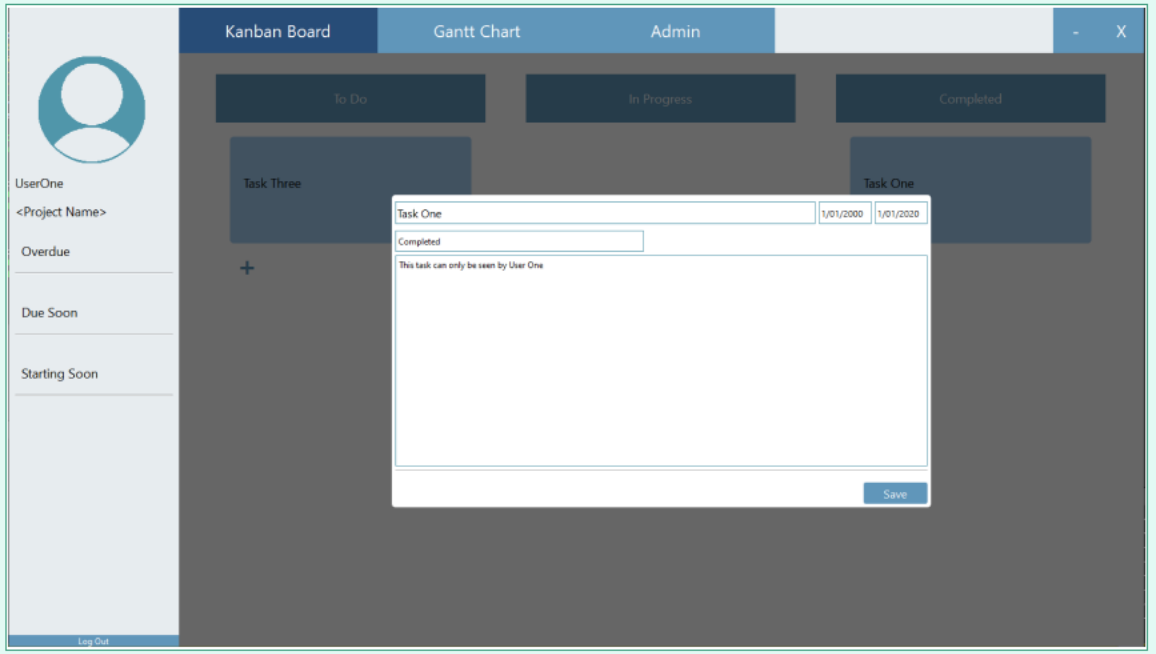
Freddie Rive

CS301 Summary

I made a Kanban Board in Qt/C++ that utilized some Zero Trust Security practices, most notably Access Control and encryption of data while in transit and at rest.

I was able to create a robust method for restricting access at the database level with NoSQL queries, preventing unauthorized data from leaving the database.

Unfortunately, I was unable to use the encryption library I had wished for as it was incompatible with Qt, forcing me to settle for an inferior alternative.



Example Screenshot of the application from CS301

MVP Project Plans

- Port application over to Vue.js
- Use a web host
- Add administrator role, with ability to:
  - Modify user access
  - Backup project, load backups
  - View Activity Logs
  - Delete Accounts
- Extra security for Admin accounts
  - Multi-factor authentication
- Gantt Chart
- Notifications

Application Development

Front-end Language: Vue.js

Back-end Language: Node.js

Database: MongoDB

WBS / Kanban: Notion

Version Control: Github

Project Manager: Rouwa Yalda

Aa Name	Tags	Date	Status
Design Log Wireframes	WIP Logs Aesthetic	August 31, 2023	Done
Test Log Wireframes	WIP Logs Aesthetic	September 3, 2023	Done
Refine Logs Display	WIP Logs Aesthetic	September 9, 2023	Done
Create Admin Wireframes	WIP Admin Aesthetic	September 2, 2023	Done
Test Admin Wireframes	WIP Admin Aesthetic	September 3, 2023	Done
Refine Admin Designs	WIP Admin Aesthetic	September 9, 2023	Done
Update Class Diagrams	WIP Technical	September 10, 2023	Done
Update Use Case Diagram	WIP Technical	September 12, 2023	Done
Create Activity Diagrams	WIP Technical	September 12, 2023	Done
Create Presentation	WIP Hand-in	September 13, 2023	Done
Film Presentation	WIP Hand-in	September 14, 2023	Done
Finish, Polish Writeup	WIP Logs Admin Paper	September 14, 2023	Done
Hand in IDD	WIP Paper Hand-in	September 15, 2023	Done
Convert Application to Vue	MVP Development Vue Technical	September 22, 2023	Done
Add Log Schema to Database	MVP Development Technical Backend Databse Logs	September 18, 2023	Done
Implement Logging System	MVP Development Technical Backend Databse Logs	September 26, 2023	Done
Create Admin Page	MVP Development Vue Admin	October 1, 2023	Done
Create Admin Access Control Functions	MVP Development Technical Vue Backend Logs	October 3, 2023	Done
Add Multi-Factor Authentication	MVP Development Vue Technical Backend	October 7, 2023	In progress

Work Breakdown Structure in Notion

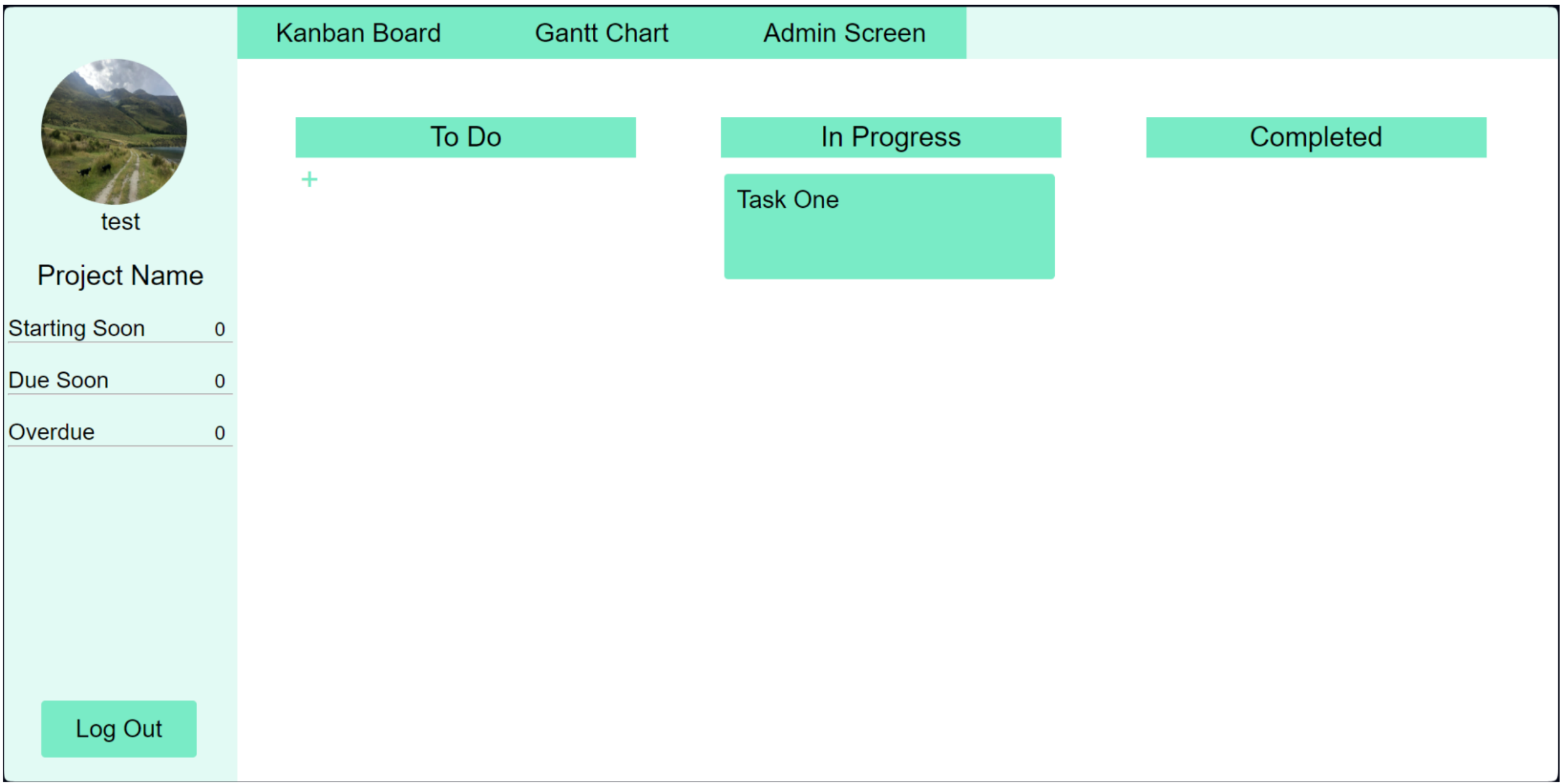
Struggles

Changing the language has resulted in a significant number of bugs. This has consistently slowed development progress as I discover new bugs within the applications basic functionality that need to be repaired.

The addition of the Admin role significantly increased the complexity of the application due to the measures taken to secure it. Since the admin will need to be sent data on all users and tasks, there have to be systems in place to ensure that only required data is transmitted to mitigate risk

While I had planned to include IP blocking in the application, the significant amount of work this would have required made it impossible to complete in the given timespan

Layout


Kanban Board
Gantt Chart
Admin Screen
test
Project Name
Starting Soon 0
Due Soon 0
Overdue 0
Log Out
To Do
In Progress
Completed
+
Task One

Administrator Role

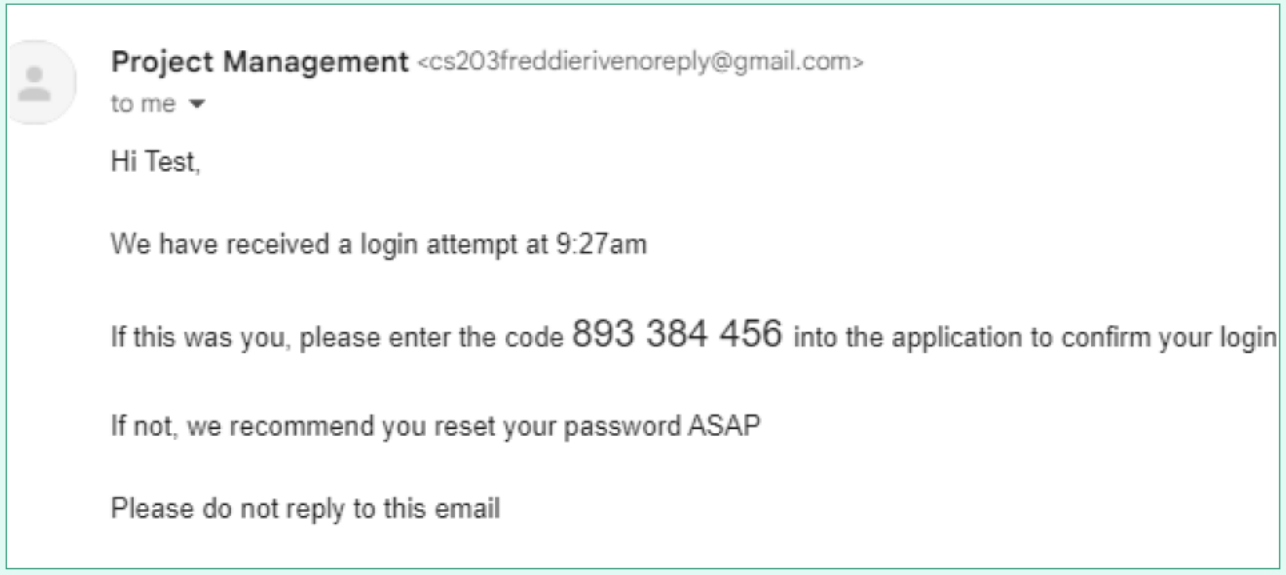
The basic functionality of the role was fairly easy to implement, as it mostly consisted of editing existing data in the database.

Since this new role naturally requires access to all the application, it became a clear security risk. In order to protect accounts with this permission, I decided to add 2-factor authentication

2-Factor Authentication

One popular method of securing accounts is using multi-factor authentication, a process where users are required to confirm their identity using more than just credentials. The most common form of multi-factor authentication is emailing or texting a private code to the user. For this app, we will be using email.

Node.JS has a module called ‘nodemailer’ that makes it easy to send emails from the backend.



Email sent automatically by the backend

Encryption

A modern encryption system is expected to encrypt data while ‘in transit’ and ‘at rest’.

The application uses CryptoJS, a cryptography library that is a notable upgrade from the one I used in CS301.

CryptoJS encryption uses Cipher Block Chaining, a process which ensures that each encryption of a given plaintext will return a different ciphertext while still being decryptable.

```
_id: ObjectId('655cb459e265df4456bbdf14')
name: "U2FsdGVkX1+aMHiDxpPS4ZZiI7pIlRPeUlGxLV/6ENo="
description: "U2FsdGVkX1+h/gTFMuOXDiRNpcLsRkTkZKI+48/YXgM="
startDate: 2023-11-06T00:00:00.000+00:00
endDate: 2023-11-09T00:00:00.000+00:00
state: "U2FsdGVkX1+J8JzdBoKm5FpyocvFmK/Dw/IoEO+CVIU="
▾ users: Array (1)
    0: ObjectId('655a0b18246914f44324c6e5')
```

Data encrypted at rest

```
{
  _id: new ObjectId("655cb459e265df4456bbdf14"),
  name: 'U2FsdGVkX1+aMHiDxpPS4ZZiI7pIlRPeUlGxLV/6ENo=',
  description: 'U2FsdGVkX1+h/gTFMuOXDiRNpcLsRkTkZKI+48/YXgM=',
  startDate: 2023-11-06T00:00:00.000Z,
  endDate: 2023-11-09T00:00:00.000Z,
  state: 'U2FsdGVkX1+J8JzdBoKm5FpyocvFmK/Dw/IoEO+CVIU=',
  users: [ new ObjectId("655a0b18246914f44324c6e5") ]
}
```

Data encrypted in transit

Thank You