Anisha Parikh
Jaden Edgar
Kyra Barnes

GitHub link: https://github.com/Security-System-Company/COSC-310-Project

Our project is a security system for an office building. It should be able to scan preauthorized NFC cards to allow specific employees in specific doors within the working hours.

For our software development cycle, we chose the waterfall model. This is because the software we are developing for the security system is one that requires all parts of a step to be completed before moving on to the next. This is a safety-critical system that required a log of up-front analysis before implementation. We need a structured, plan-driven approach to our development. To maintain a high level of security the requirements should be well understood at the beginning of the project and should only have minimal changes made to it. We also want the requirements to remain as tight as possible. Because of this lack of change it is easy to complete this system in a very step by step process. This also ensured everything necessary is complete before moving on.

The steps of the waterfall cycle and how we proceeded with the system are as follows:

REQUIREMENTS

- Researching existing items
  - Real Life
    - University
      - Campus housing doors and how they work
      - Classroom and lecture hall doors
  - Media
    - Office buildings
      - Emergency situations
      - Visitors
      - High level access
  - Research
    - Case study - MentCare
      - Security memory/logs
- Deciding which items from each
  - Using NFC cards like the key cards used on campus
  - Having specific visitor cards
  - Creating a similar system to MentCare's activity logging system
  - Emergency measures to ensure doors do not remain locked when quick exist is necessary
  - Quick response time
- Narrowing down requirements to what is practical and within our capabilities

SYSTEM AND SOFTWARE DESIGN

- History

  - Assess the languages and coding software we know
    - Java (processing, android studio), python, SQL
- NFC
  - Options that would allow us to stimulate NFC cards tapping on a door
    - Android studio would allow for emulating and assigning cards
- Visuals
  - Which software is best suited for this situation
    - Processing
- Assign roles based on ability
  - Split responsibility based on how the tasks can divide
    - NFC cards
    - Visual Representation
    - Project Management

IMPLEMENTATION AND UNIT TESTING

- NFC/Phone App
  - Android studio (Java)
  - Research for implementation
  - Implemented basic features of the android studio app for user functionality
  - Implemented NFC scanning capabilities of android phone
- Visual/Website
  - Processing (Java)
  - Made an office layout using online software
  - Resized the software to necessary specs
  - Added visuals to processing
  - Coded doors different colours to represent when they open and close
  - The doors stay open (green) for three seconds and then close (red)
- Testing
  - Basic functions for both NFC and Visual aspects to later expand to connect the two

INTEGRATION AND SYSTEM TESTING

- Register cards with app
  - Find cards
  - Connect to app
- Add a networking interface to allow for phone app to communicate with software to emulate the situation
  - Find interface
  - Connect it to the phone app
  - Connect to secondary software
- Connect the two separate programs
- Test various situations
  - Safety measures
  - Doors opening
  - Doors failing to open
  - Logging interactions

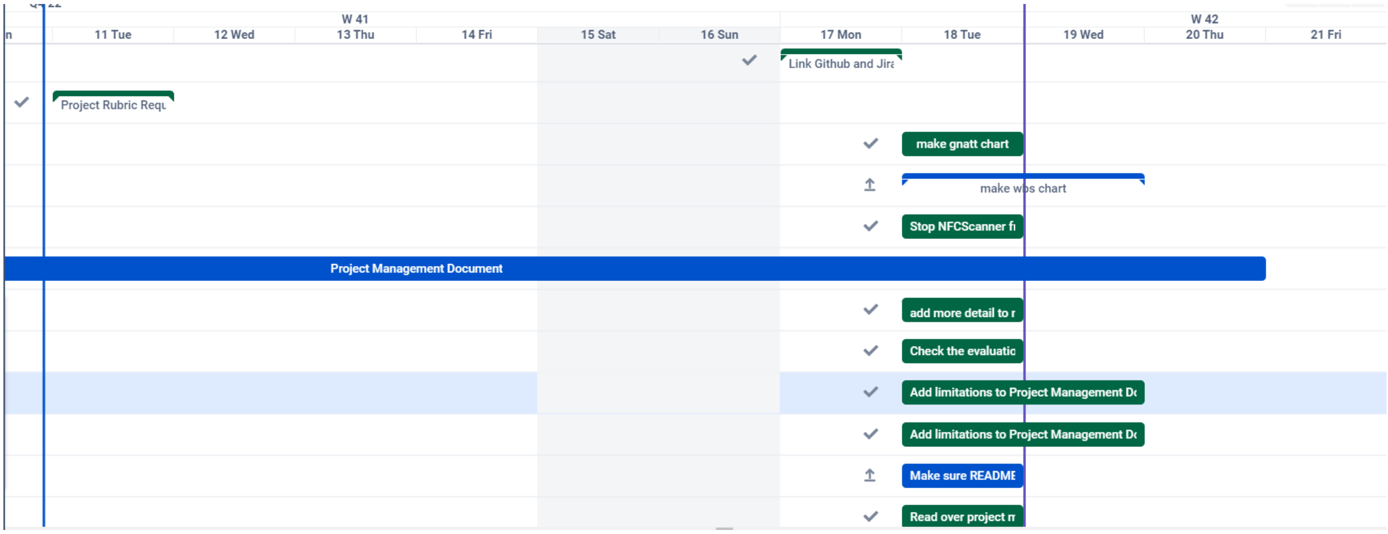
OPERATION AND MAINTENANCE

- Find holes in testing
- Make sure no new issues arise
- Make sure code is up to date as outside software (Processing and Android Studio) are updated

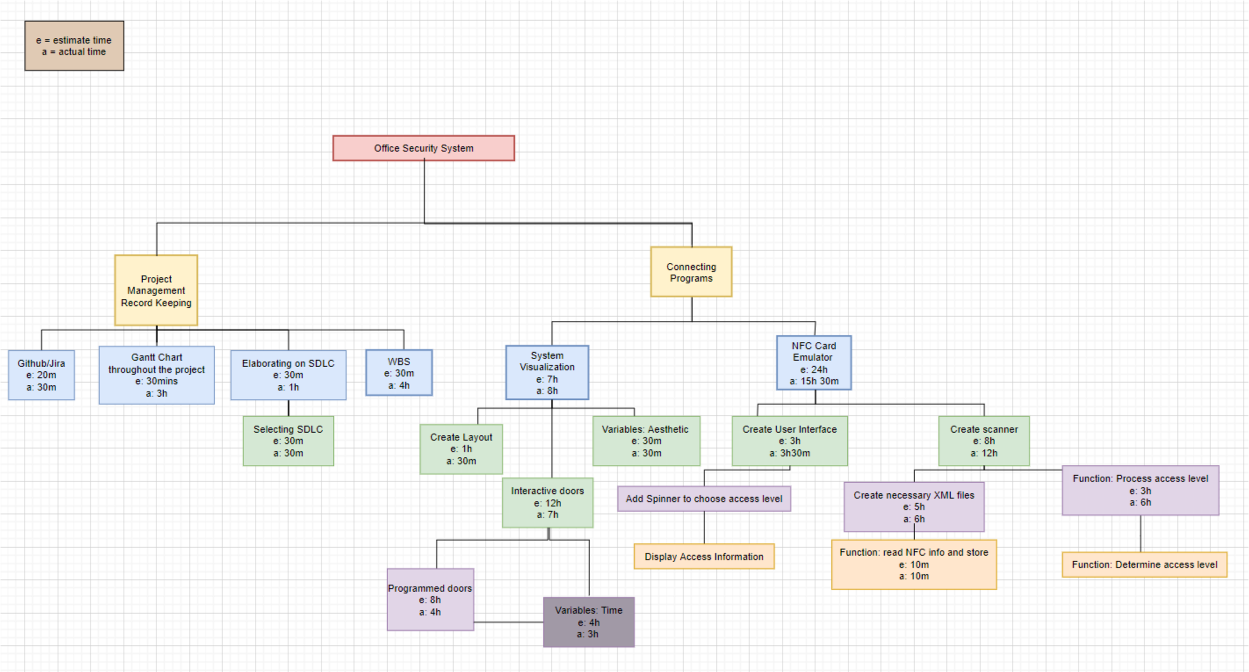
LIST OF LIMITATIONS

- We cannot emulate someone holding open a door or any other suspicious activity with our program.
- Our software can currently only emulate single person use.
- App refreshes every time the card is scanned.
- Currently do not have networking solution implemented to allow processing java application and android studio application to communicate with each other.
- We don't have a database setup to store specific information regarding door activity.

GANTT CHART



This is a portion of our Gantt chart. We added tasks as we went jus to ensure that the tasks and how long they took were accurate. The tasks are broken down similarly to the WBS, in the sense that each project member has their major task and made smaller "child tasks" based on what needed to be done to complete the general task. This Gantt chart does go more into depth and more specific jobs than the WBS as we added every issue, we ran into that needed to be resolved to keep our teammates updated. This was used as a form of communication


e = estimate time
a = actual time
Office Security System
Project Management Record Keeping
Connecting Programs
Github/Jira e: 20m a: 30m
Gantt Chart throughout the project e: 30mins a: 3h
Elaborating on SDLC e: 30m a: 1h
WBS e: 30m a: 4h
Selecting SDLC e: 30m a: 30m
System Visualization e: 7h a: 8h
NFC Card Emulator e: 24h a: 15h 30m
Create Layout e: 1h a: 30m
Variables: Aesthetic e: 30m a: 30m
Interactive doors e: 12h a: 7h
Programmed doors e: 8h a: 4h
Variables: Time e: 4h a: 3h
Create User Interface e: 3h a: 3h30m
Create scanner e: 8h a: 12h
Add Spinner to choose access level
Display Access Information
Create necessary XML files e: 5h a: 6h
Function: read NFC info and store e: 10m a: 10m
Function: Process access level e: 3h a: 6h
Function: Determine access level

The WBS chart is a very general breakdown of the different work assignments and how long each of them took. It is also discreetly divided up by who did what. Anisha was responsible for most things under “Project Management Record Keeping”, Kyra worked on everything under “System Visualization” and “GitHub/Jira”. Jaden worked on the “NFC Card Emulator” category. We all collaborated and helped each other where necessary.