RealTrak Train Delay and Scheduling System
CSCI 524 Analysis & Design of Software Systems

Krystal Jordan
Nicolo Dannemiller
Paul Stevens
Palash Joshi

Version <3.0>

Revision History

Date	Version	Description	Author
4/7/2021	1.0	Initial system analysis on RealTrak system including the system goals and scope. Identifying stakeholders, functionality, and non-functional requirements has been completed. User interface requirements discussions have led to an initial design of the user interface.	Krystal Jordan Nicolo Dannemiller Paul Stevens Palash Joshi
4/11/2021	2.0	User and use cases have been identified and modeled using user stories and event decomposition techniques. Use case descriptions and system sequence diagrams completed. Domain model and class diagram are in place showing relationships for each class and functions used. Database schema identified and in place. Working user interface has been drawn up.	Krystal Jordan Nicolo Dannemiller Paul Stevens Palash Joshi
4/22/2021	3.0	Final project version completed. The project problem statement was finalized. The project plan and scheduling was recorded and a first iteration schedule was created. Our implementation and debugging procedures are now stated in the proper sections. Completion of final summary of the project and any sections of the appendix were finished by the appropriate team members.	Krystal Jordan Nicolo Dannemiller Paul Stevens Palash Joshi

Table of Contents

Project Description

1.Introduction

The RealTrak train delay and scheduling system is a system that is designed to help train passengers receive accurate information about the arrival and departure times, including any delays, of trains at stations around the city. Accurate reporting of arrival, departure, and delay times is a crucial part of any mode of public transportation because it's what determines if a passenger is going to arrive at their destination when they need to be there. Third-party transit status tracking software, such as the Long Island Rail Road app, are able to provide users with real-time data on trains based on the integration of estimated time of arrivals (ETAs) that are provided by the railway. The resulting analysis and design of this train delay and scheduling system will give passengers at each train station and any third-party tracking software updates on delays and real-time arrival and departure times. (Paul)

1.1.Overall goals of the system

The intention of this tracking and prediction system is to provide railway employees an easy way to track trains by logging departure times and any delays that a train may incur along its route. The delays will automatically update the arrival time of the train to the stations on its route. The system goal is to increase accuracy of train times and confidence of passengers in the railway system by providing ease of use to the railway employees and real-time updates on trains to passengers at each station and to third-party transit status tracking software. (Paul)

1.2.Scope of project

Our team is in charge of analyzing current business functions and operations, designing a functional system that meets the business needs, and developing a working solution that provides accurate times of each train. The system will be used in an urban setting by a set of railways connected by train stations covering an area of 25 square miles. The railways employees will access the system using a desktop application which features a graphical user interface (GUI). The train information is made available to the passengers through a web interface which can be accessed from either a desktop or mobile web browser. Each train station also features digital scheduling boards that provide passengers at the stations the same real-time train information. Third-party tracking and status software will be able to pull information about each train and its status from the system's Web API. (Paul)

1.3.Definitions, Acronyms and Abbreviations

See Appendix A.4 for Definitions, Acronyms, and Abbreviations

1.4.References

See Appendix A.5 for References

2.Project Problem Statement

Train and rails delays are a massive headache for all involved. Delays can have a negative impact on civil and commercial users. Although there is little that can be done to eliminate a delay once it has occurred, quick, accurate, and early reporting of a delay can allow all users involved to plan accordingly and react appropriately. (Nicolo)

2.1.System Vision

2.1.1.Problem Description

Delays are a regular, mostly random set of issues that can cause the train, line or station to lose its ability to adhere to a set schedule. These delays can disrupt travel, infrastructure, commerce and even national defense. These delays are often hard to communicate and slow to be reported to all affected persons. (Nicolo)

2.1.2. System Capabilities

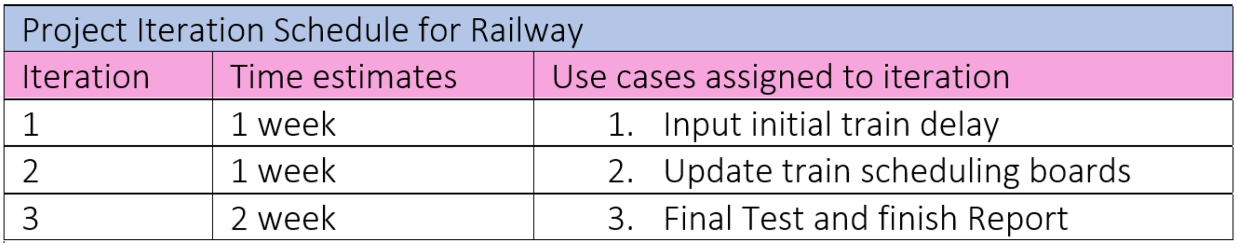
The system should reliably and automatically update all stations, and schedules upon submission by employees. The human element is still needed to analyse and calculate the nature, cause, and length of the delay.

2.1.3. Business benefits

By investing in a fully connected self propagating delay system customers of all forms, civil, industrial, commercial, and federal, can hold confidence that the company will keep them updated and informed on the status of their concerns. (Nicolo)

3.Project Plan and Monitoring

3.1.Project Iteration Schedule



Project Iteration Schedule for Railway		
Iteration	Time estimates	Use cases assigned to iteration
1	1 week	1. Input initial train delay
2	1 week	2. Update train scheduling boards
3	2 week	3. Final Test and finish Report

Figure Project Iteration Schedule

3.2.Work Breakdown Structure

3.2.1.Project planning

a. Identify the problem: Delays and proper recording of delays in the train scheduling system is a common problem being faced by railway corporations and other stakeholders like customers and partner organizations. Our project/system brings this main problem into consideration and presents an effective solution that would cover all aspects of this problem - 1 day.
b. Get an approval for the project - 3 days.
c. Discussing and finalizing on the work roles and responsibilities and time limit of the project - 3 days.
d. Setting up the work environment and working on the documents and procedures that

need to be followed - 3 days.

e. Scheduling the work using the project iteration schedule - 3 days.

3.2.2.Analysis tasks

a. Identifying the users and other stakeholders related to the project - 4 days.
b. Identifying functional and non-functional requirements from users and stakeholders by means of information gathering techniques such as distributing questionnaires among them. Questionnaires will be related to customers who board trains on a daily basis or railway employees who work on scheduling systems - 4 days.
c. Modeling system activities using UML diagrams - 7 days.

3.2.3.Design tasks

a. Design database schema and include necessary data for the database - 3 days.
b. Design user interface layout - 9 days.
c. Identify and diagrammatically explain user classes, attributes and methods through class diagrams - 7 days.

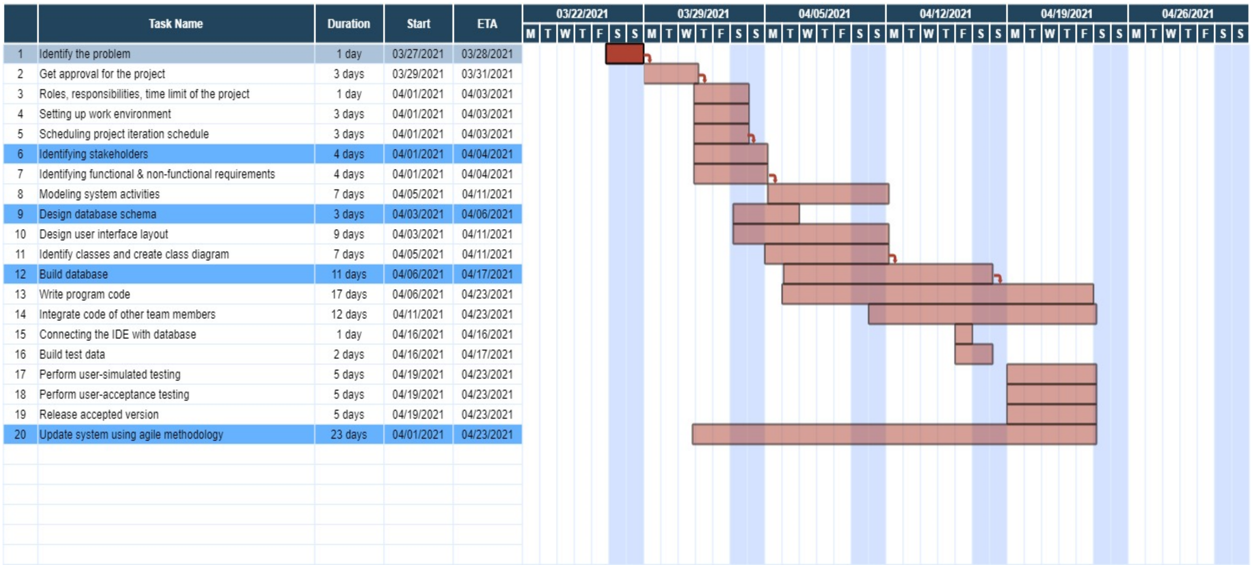
3.2.4.Build tasks

a. Build database - 11 days.
b. Write program code - 17 days.
c. Integrate code written by other team members - 12 days.
d. Connecting the IDE with the database - 1 day.
e. Build test data - 2 days.
f. Perform user-simulated testing - 5 days.
g. Perform user-acceptance testing - 5 days.
h. Release accepted version - 5 days.
i. Update the system using agile methodology - 23 days.

3.2.5.Notes

a. First iteration deals with gathering information on stakeholders and modeling them into requirements for the system. First iteration also focuses on the preliminary user interface which presents a graphical presentation of the system. Rest of the iterations, which include designing and building the system will be performed on a parallel approach in a deliverable manner by team members Krystal, Nicolo, and Paul.
b. This is a four person team. The work pattern involves (1) identifying the problem and fact-finding by Paul; (2) creating a system vision by Nicolo; (3) project monitoring by Krystal and Palash; (4) system analysis by all team members; (5) system design, implementation, and testing by Nicolo, Krystal, and Paul.
c. The use case that will be developed during this iteration is employee scheduling a delay on the train scheduling subsystem.

3.3.First Iteration Schedule



	Task Name	Duration	Start	ETA
1	Identify the problem	1 day	03/27/2021	03/28/2021
2	Get approval for the project	3 days	03/29/2021	03/31/2021
3	Roles, responsibilities, time limit of the project	1 day	04/01/2021	04/03/2021
4	Setting up work environment	3 days	04/01/2021	04/03/2021
5	Scheduling project iteration schedule	3 days	04/01/2021	04/03/2021
6	Identifying stakeholders	4 days	04/01/2021	04/04/2021
7	Identifying functional & non-functional requirements	4 days	04/01/2021	04/04/2021
8	Modeling system activities	7 days	04/05/2021	04/11/2021
9	Design database schema	3 days	04/03/2021	04/06/2021
10	Design user interface layout	9 days	04/03/2021	04/11/2021
11	Identify classes and create class diagram	7 days	04/05/2021	04/11/2021
12	Build database	11 days	04/06/2021	04/17/2021
13	Write program code	17 days	04/06/2021	04/23/2021
14	Integrate code of other team members	12 days	04/11/2021	04/23/2021
15	Connecting the IDE with database	1 day	04/16/2021	04/16/2021
16	Build test data	2 days	04/16/2021	04/17/2021
17	Perform user-simulated testing	5 days	04/19/2021	04/23/2021
18	Perform user-acceptance testing	5 days	04/19/2021	04/23/2021
19	Release accepted version	5 days	04/19/2021	04/23/2021
20	Update system using agile methodology	23 days	04/01/2021	04/23/2021

4.System Analysis

4.1.Stakeholders

4.1.1.Internal Stakeholders:

Rail employees, Internal Auditors, Operational Managers and Board of Directors

4.1.2.External Stakeholders:

Customers, External auditors, Investors and regulators.

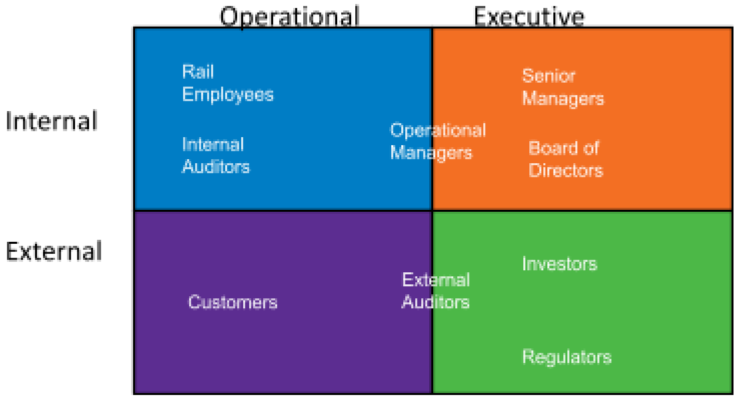
4.1.3.Operational Stakeholders:

Rail Employees, Internal Auditors, Customers, Operational Manager, and External Auditors

4.1.4.Executive Stakeholders:

Senior Managers, Board of Directors, Investors External Auditors, and Regulators

4.1.5.Stakeholder Diagram



4.2.Questionnaires

Railway Questionnaire

This questionnaire is being sent to all railway personnel. As you know, Railway [company] is developing a new train delay system to update customers and railway employees of train delays.

The purpose of this questionnaire is to obtain preliminary information to assist in defining the requirements for the new system. Follow-up discussions will be held to permit everybody to elaborate on the system requirements.

Part I. Answer these questions based on a typical 8-hour shift.

1. What zone of the railway system do you service?____
2. Estimate how many trains pass your zone during your shift?_________________________________
3. Estimate how many trains are delayed that pass your zone during your shift?___________________
4. On average, by how many minutes are the trains delayed?__________________________________
5. Approximately how many times does a customer ask if a train has been delayed? _______________

Part II. Circle the approximate number on the scale from 1 to 7 based on how strongly you agree or disagree with the statement.

Question	**Strongly Agree**						**Strongly Disagree**
It would save railway employees time if the information screens automatically entered train delay times.	1	2	3	4	5	6	7
It would improve customer satisfaction if the information screens automatically entered train delay times.	1	2	3	4	5	6	7
I could better perform my job satisfaction if the information screens automatically entered train delay times.	1	2	3	4	5	6	7

Part III. Please enter your opinions and comments.

Please briefly identify the problems with the current system that you would like to see resolved in a new system.

____

____

____

____
____
____

Customer Questionnaire (WIP)

Part I. Answer these questions based on your personal experience.

1. What is your main use for the <NAME OF RAIL>? Personal, commute, business, commerce, freight, other ______
2. Estimate how many times a week do you use the rail's services? ________________________________
3. Estimate how many train delays do you encounter on an average day?__________________
4. On average, by how many minutes are the trains delayed?_________________________________
5. What steps do you need to take to find out if your train has been delayed?_______________

Part II. Circle the approximate number on the scale from 1 to 7 based on how strongly you agree or disagree with the statement.

Question	**Strongly Agree**						**Strongly Disagree**
Delay reporting is currently inefficient.	1	2	3	4	5	6	7
The lack of an efficient system has cost me time, money, and/or sanity.	1	2	3	4	5	6	7
I believe the new system will greatly improve my usage of the trains.	1	2	3	4	5	6	7

Part III. Please enter your opinions and comments.

Please briefly identify the problems with the current system that you would like to see resolved in a new system.

____
____
____

____

____

____ (Nicolo)

4.3.Functionality

The new railway set of systems will allow rail authorities, engineers, and admins, to rapidly update and adjust train departures, schedules, and delays to keep commuters up to date.

4.4.Non-functional requirements

4.4.1.Usability

Screens at the railway that update the passengers of the train departures, schedules, and delays. User Interface for railway employees to make updates to the train departures, schedules, and delay in the new system. (Krystal)

4.4.2.Reliability

The new system will exhibit incorrect processing or outages less than 2% of the time. (Krystal)

4.4.3.Performance

The update of a delayed train by a rail employee will have a one-second response time to pressing the submit button and the update of the screens. (Krystal)

4.4.4.Security

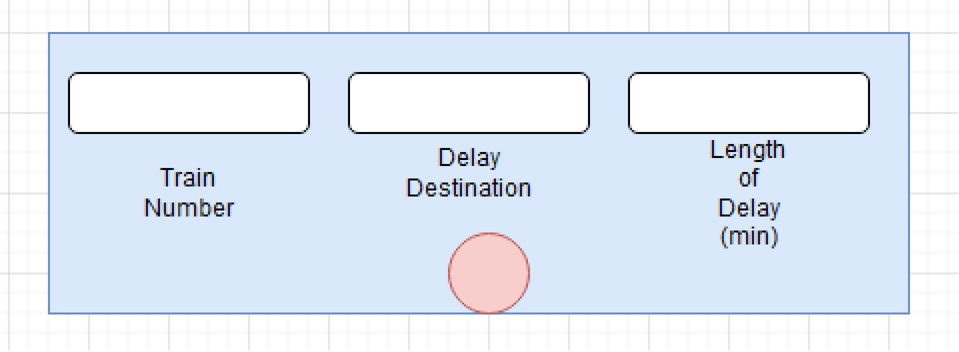
No customer can edit delays, train information, or destinations. Only management and upper level employees should be able to edit train information. Delays and destinations can only be updated by conductors, engineers, management, and operators. (Nicolo)

4.4.5.Other requirements

Third-party train status and tracking apps should be able to interface with the system to provide real-time information on each train. (Paul)

4.5.Interfaces

Sample of employee facing interface:



The three text fields allow for employees to input the relevant information. Train number should be optional to allow for station wide delays. Delay destination should also be optional in cases where the train is delayed regardless of station (for example derailment.) Additional options should include multiple entries for trains if some but not all trains at destination are delayed, or if an incident involves multiple trains.

Train number should print out the name of inputted train for extra verification and avoidance of misplaced delays. For example as a user types in train number 1875 the UI should display under the text field, the train's name "Ferdinand Porsche". This way the user inputting the delay has a way to verify that they are assigning the correct train the delay. If a user mistypes the train number the addition of the train name will help users identify their mistake.

Delay destination should have a suggestion list for faster selection of potentially long/complicated names. A drop down menu could work, but would not scale well with large interconnected systems. Combination of drop menu and suggestion function would be best but might require more time than our scope allows for.

Length of delay should be a whole number that represents the expected length of the delay. There are too many factors to account for in most delay events that would prevent a prediction of delay accurately enough to include the seconds. Length of delay should **not** be optional, if the delay is indefinite or unknown, delay length field should be capable of taking timed and untimed values to account for known and unknown delay lengths. A check box for "Delay Length Unknown" would work to fix, or having the field be able to accept strings and integer inputs. In the string case, the system should remove the time field from the train display and add the imputed string in the "train status" field. For integer cases the system should take the numeric input and add it to the current expected arrival time to all affected stations. (Nicolo)

4.6. User Stories and Use Cases

4.6.1. User Stories

User Story 1
As a train station employee, I want to easily update any delays in the arrival times of a train so that the arrival times are displayed shortly after delays are entered.

Acceptance Criteria:

1. Easy-to-navigate employee GUI
2. Suggestions made to employee on which trains are currently running
3. System prompt showing details of delay
4. System prompt showing success or failure of a delay entry

User Story 2
As a system administrator, I want to make sure that only the appropriate employees are able to log into the system to make a change in a train's delay time.

Acceptance Criteria:

1. Set up access control for the system based on business requirements and who needs access to the system
2. Detailed logs on who accesses the system and what entries are made by each railway employee
3. Quarterly review of business practices regarding system access

4. Educate on proper password guidelines

User Story 3
As a passenger, I want train arrival times to be accurate and easily accessible so that I can get to my destination on time.

Acceptance Criteria:

1. Delays made through the system must update to the arrival or departure time of a train then display it properly at each train station scheduling board
2. Audible announcements should be made at each station a train stops at if a delay is entered for that train
3. Railway employees should perform proper maintenance on scheduling boards and announcement equipment to ensure operational effectiveness

(Paul)

4.7. Use Case Models

4.7.1. Users and their Goals

User	User goal and resulting use case
Railway employee	Login to system Enter new train delay Update existing train delay Remove a train delay
System administrator	Grant system access to employee Revoke system access to employee Check system logs
Passenger	Check train arrival/departure status

(Paul)

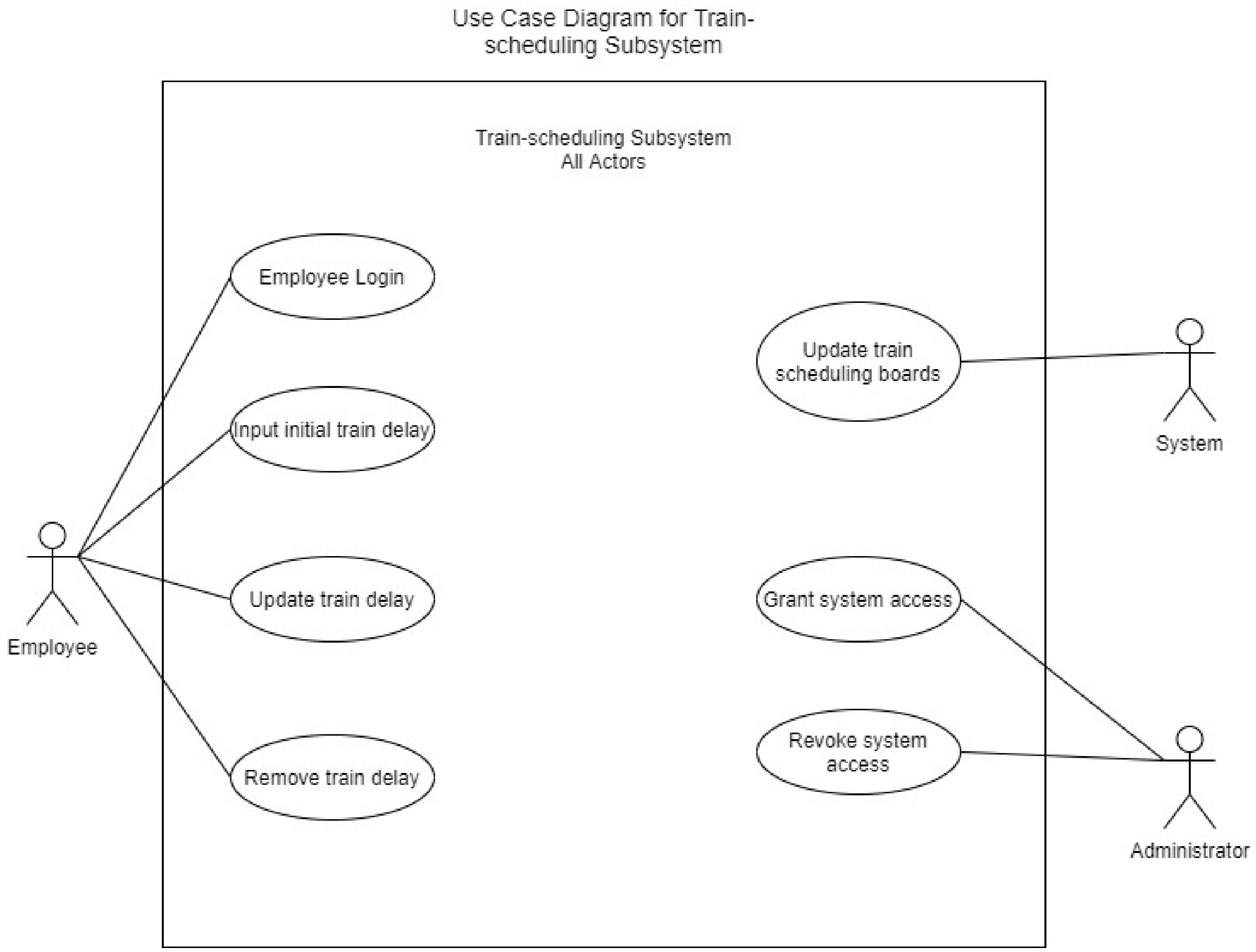
4.7.2. Use Cases

Event	Type	Use Case	Actor
Employee logs into system	External	Employee login	Employee
Employee inputs initial delay of a train	External	Input initial train delay	Employee
Employee updates time of delay on a train with existing delay	External	Update train delay	Employee

Employee removes delay when train is now on time	External	Remove train delay	Employee
System updates train scheduling boards with new train arrival/departure times when a delay has been logged	Temporal	Update train scheduling boards	System
Grant employee access to delay system	External	Grant system access	Administrator
Revoke employee access to delay system	External	Revoke system access	Administrator

(Paul)

4.7.3.Use Case diagram(s)


Use Case Diagram for Train-scheduling Subsystem
Train-scheduling Subsystem
All Actors
Employee Login
Input initial train delay
Update train delay
Remove train delay
Update train scheduling boards
Grant system access
Revoke system access
Employee
System
Administrator

4.7.4.Use Case Descriptions

<table>
<tr><td><b>Use case name:</b></td><td colspan="2">Input initial train delay</td></tr>
<tr><td><b>Scenario:</b></td><td colspan="2">An employee would like to input a delay for a train into the system</td></tr>
<tr><td><b>Triggering event:</b></td><td colspan="2">A train has experienced a delay and will no longer arrive or depart at the scheduled time</td></tr>
<tr><td><b>Brief description:</b></td><td colspan="2">To ensure that passengers have accurate times for arrival and departure of trains, any delay that a train experiences during its route must be input into the system so stops that it has yet to visit will adjust arrival and departure times accordingly.</td></tr>
<tr><td><b>Actors:</b></td><td colspan="2">Employee</td></tr>
<tr><td><b>Related use cases:</b></td><td colspan="2">Update train scheduling boards</td></tr>
<tr><td><b>Stakeholders:</b></td><td colspan="2">Employee, Passenger, Operational Manager</td></tr>
<tr><td><b>Preconditions:</b></td><td colspan="2">A train and its route must exist in the database to assign a delay to it</td></tr>
<tr><td><b>Postconditions:</b></td><td colspan="2">After a successful delay input, updates on arrival and departure times are sent to the proper train scheduling boards to display new times</td></tr>
<tr><td rowspan="2"><b>Flow of activities:</b></td><td><b>Actor</b></td><td><b>System</b></td></tr>
<tr><td>1. Employee logs in to system

2. Train and delay information is entered into the correct fields and is submitted</td><td>1.1 System GUI is presented to the employee

2.1 Verify valid train number
2.2 Verify valid destination
2.3 Confirm delay has been entered successfully</td></tr>
<tr><td><b>Exception conditions:</b></td><td colspan="2">1.1 Employee uses incorrect credentials to log in
1.2 Employee is not authorized to use the system
2.1 Train information entered is not valid
2.2 Destination entered is not valid</td></tr>
</table>

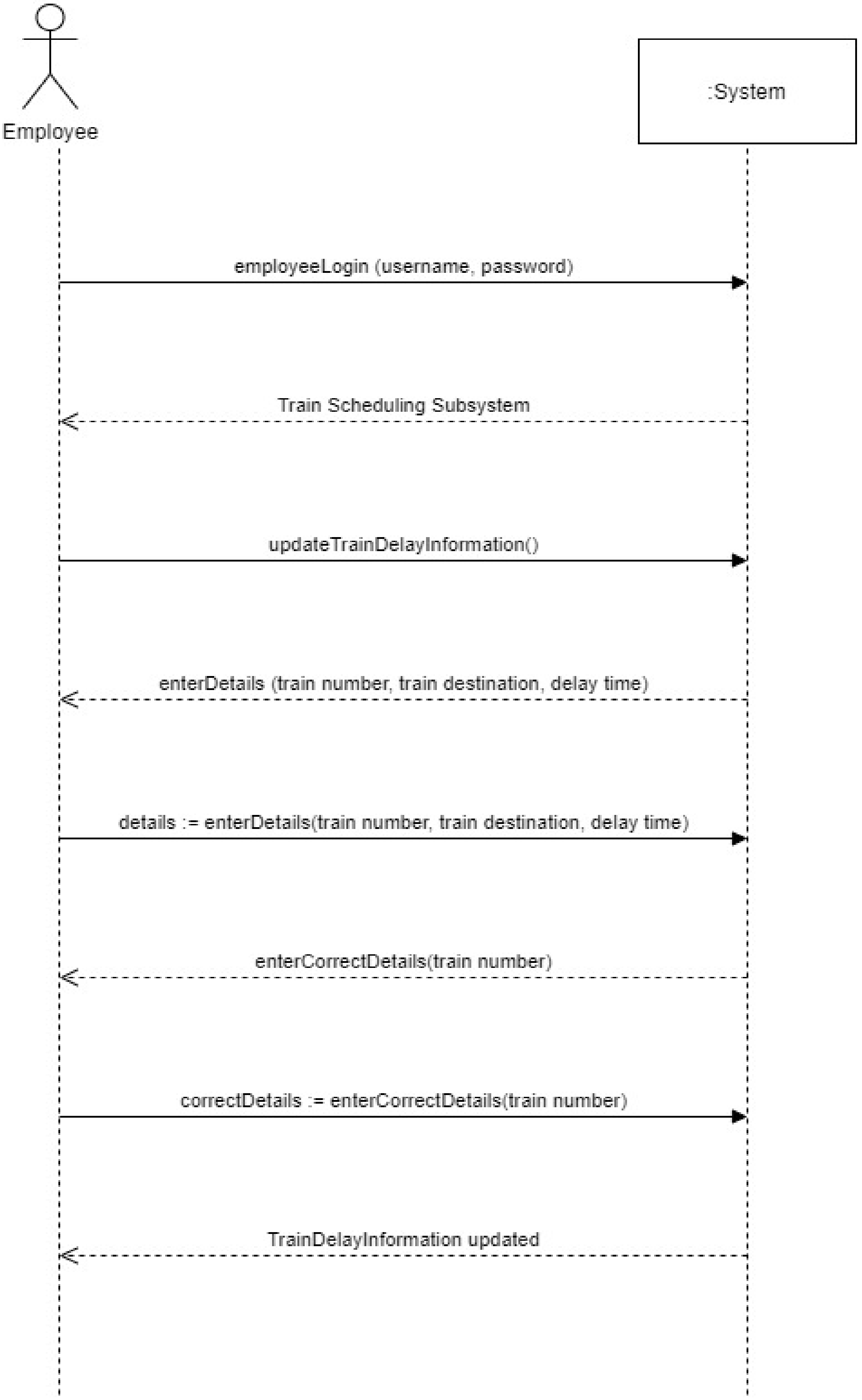
<table>
<tr><td><b>Use case name:</b></td><td colspan="2">Update train scheduling boards</td></tr>
<tr><td><b>Scenario:</b></td><td colspan="2">A change in the arrival or departure time of a train must be made available to all passengers</td></tr>
<tr><td><b>Triggering event:</b></td><td colspan="2">A train’s arrival or departure time has changed due to the input of an initial delay, a change in the delay, or the removal of a delay.</td></tr>
<tr><td><b>Brief description:</b></td><td colspan="2">To ensure passengers have accurate train arrival and departure times, the system should update the train scheduling boards at each train station soon after a delay is entered.</td></tr>
<tr><td><b>Actors:</b></td><td colspan="2">System</td></tr>
<tr><td><b>Related use cases:</b></td><td colspan="2">Input initial train delay
Update train delay
Remove train delay</td></tr>
<tr><td><b>Stakeholders:</b></td><td colspan="2">System, Passenger, Operational Manager</td></tr>
<tr><td><b>Preconditions:</b></td><td colspan="2">A train is currently running
A delay has been input, updated, or removed on said train</td></tr>
<tr><td><b>Postconditions:</b></td><td colspan="2">Corrected arrival and departure times of a train are reflected in the scheduling display boards at each train station</td></tr>
<tr><td rowspan="2"><b>Flow of activities:</b></td><td><b>Actor</b></td><td><b>System</b></td></tr>
<tr><td>1. An initial delay, an update to a delay, or the removal of a delay has been made to a train</td><td>1.1 The delay time has been properly added to or removed from the scheduled arrival or departure time
1.2 The updated times are relayed to the scheduling display board
1.3 The scheduling board displays the updated times</td></tr>
<tr><td><b>Exception conditions:</b></td><td colspan="2">1.1 The wrong delay time has been input or the premature removal of a delay has been entered into the system causing the wrong time to be displayed on the scheduling board
1.2 Connection between the system and the scheduling display</td></tr>
</table>

	boards have malfunctioned

(Paul)

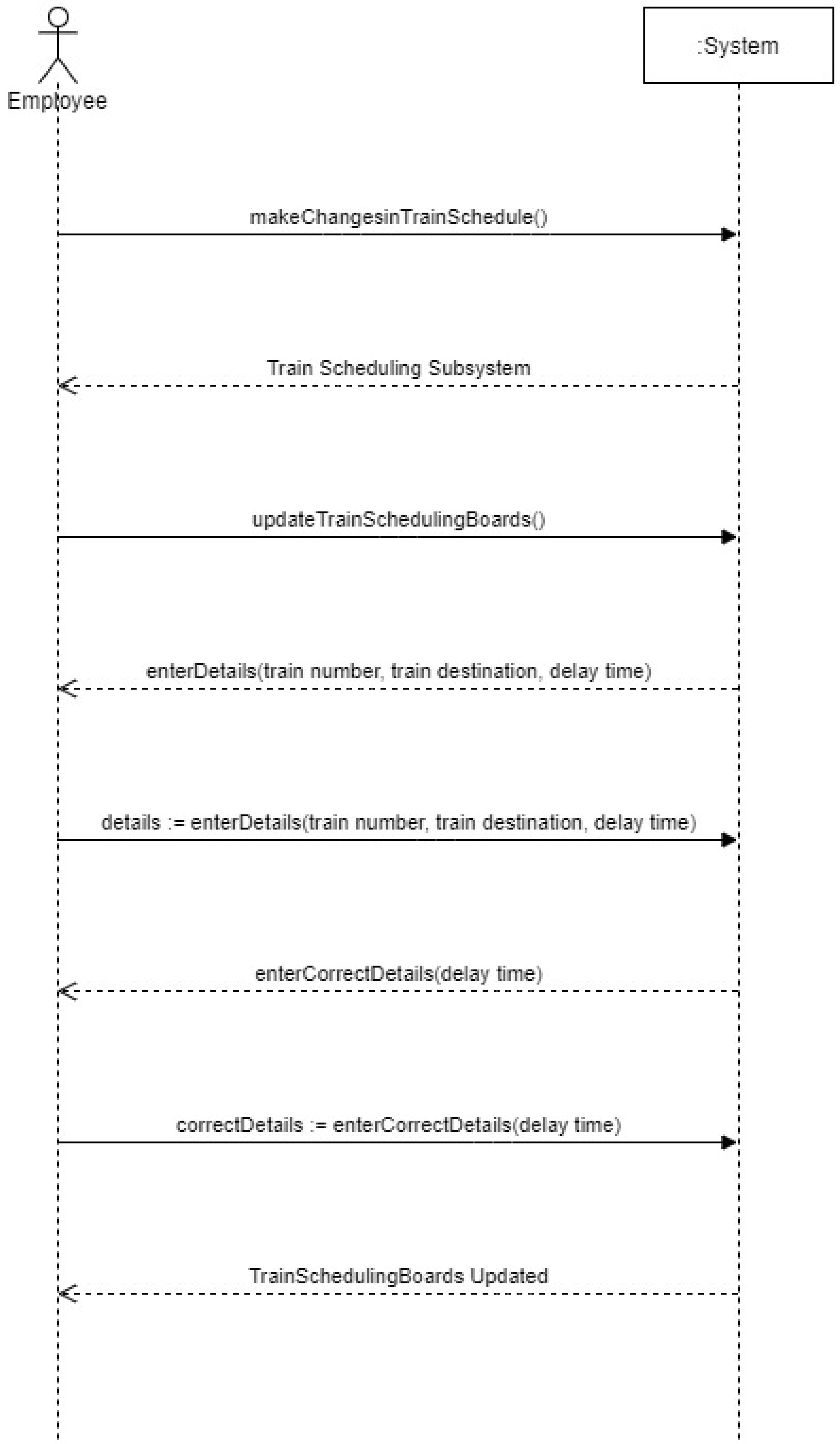
4.7.5. *System Sequence Diagram*

System Sequence Diagram for use case "Input initial train delay"



Employee
:System
employeeLogin (username, password)
Train Scheduling Subsystem
updateTrainDelayInformation()
enterDetails (train number, train destination, delay time)
details := enterDetails(train number, train destination, delay time)
enterCorrectDetails(train number)
correctDetails := enterCorrectDetails(train number)
TrainDelayInformation updated

System Sequence Diagram for use case "Update train scheduling boards"

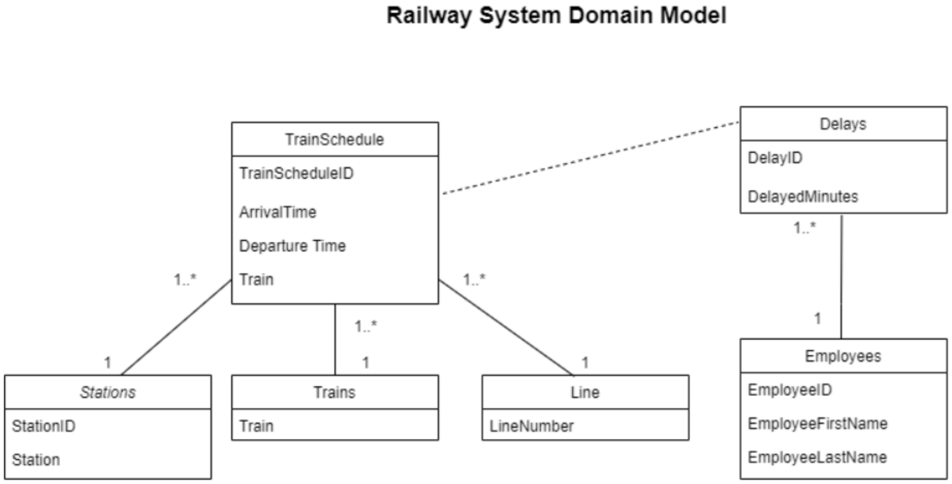

Employee
:System
makeChangesinTrainSchedule()
Train Scheduling Subsystem
updateTrainSchedulingBoards()
enterDetails(train number, train destination, delay time)
details := enterDetails(train number, train destination, delay time)
enterCorrectDetails(delay time)
correctDetails := enterCorrectDetails(delay time)
TrainSchedulingBoards Updated

4.8.Domain Class Models

4.8.1.Class Table

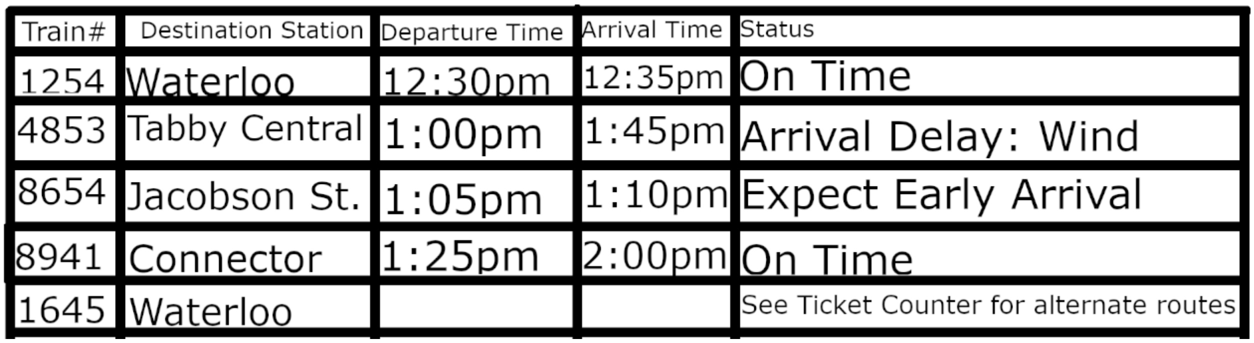
Identified noun	**Notes on including noun as a thing to store**
Train	Train number, line, stations, stop times
Line	Line description
City stop list	Locations trains stop at
Arrival time	Arrival time for trains at stations
Departure time	Departure time for trains at stations
Delay	Delay in minutes for train and its stations
Customers	They don't interact with the system just observe it, no need to store it
Railway employees	EmployeeID, and other relevant information
System administrator	We know who they are, no need to store it
Scheduling boards	An output produced by information about train delays, no need to store it

4.8.2.Domain Model Class Diagram



5.System Design

5.1.User Interface Design

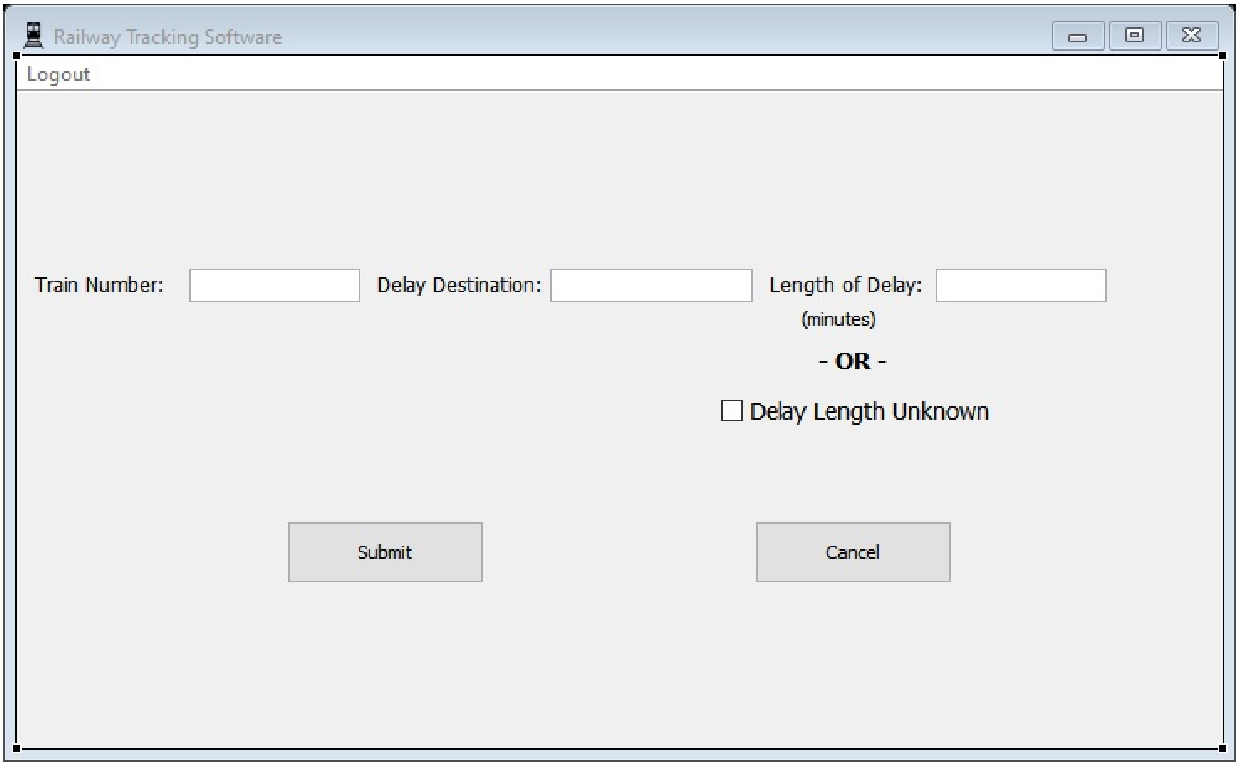


Train#	Destination Station	Departure Time	Arrival Time	Status
1254	Waterloo	12:30pm	12:35pm	On Time
4853	Tabby Central	1:00pm	1:45pm	Arrival Delay: Wind
8654	Jacobson St.	1:05pm	1:10pm	Expect Early Arrival
8941	Connector	1:25pm	2:00pm	On Time
1645	Waterloo			See Ticket Counter for alternate routes

Sample design of the train station board.

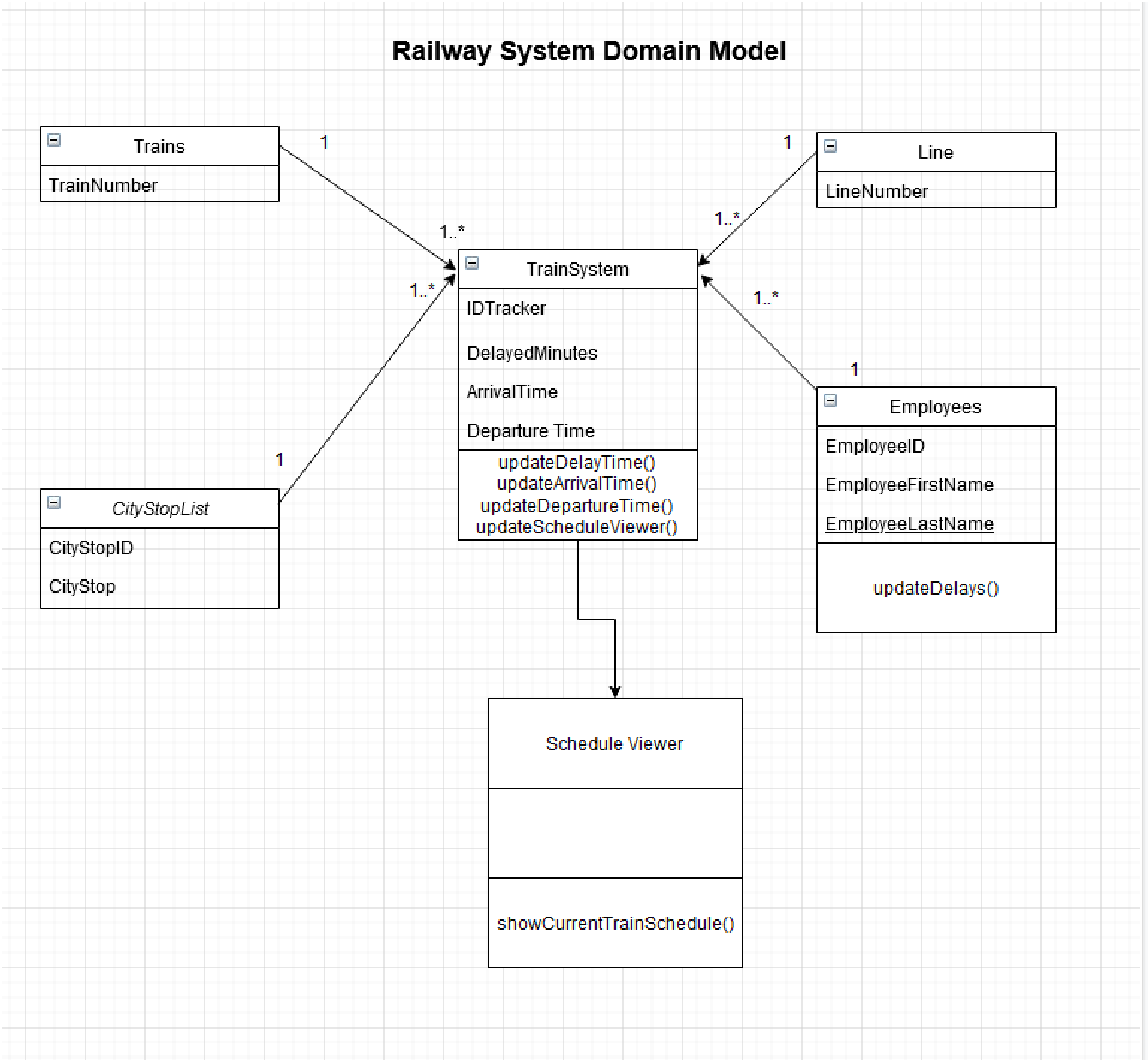
Delay system will either send updates to the board directly or update the database tables being pulled for the data. This will depend on how the existing system functions.

Four cases depicted above: On time arrivals, definite delay with reason given, possible early arrival with no reason given, and an indefinite delay with instructions for customers of that line. In this scenario all trains are leaving the same station, this is presented in the form of the board being a physical display inside the station. For off site updates and declarations source and destination stations must be listed to limit confusion as much as possible. (Nicolo)



Sample user interface for employees

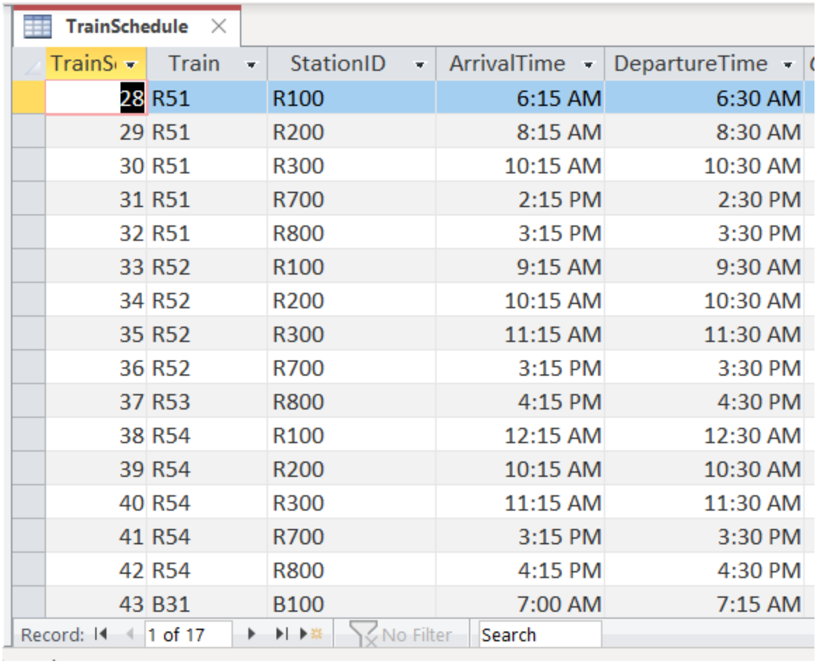
5.2.Class Diagram Design for the Entire Project



5.3.Database Design and Data

Table	Attributes
Delays	**DelayID**, LineID, EmployeeID, DelayMinutes, DelayDiscription
Employees	**EmployeeID**, EmployeeLastName, EmployeeFirstName
Lines	**LineID**
Station/Line	**Station/LineID**, StationID, LineID
Stations	**StationID**, Station
Trains	**Train**, LineID
TrainSchedule	**TrainScheduleID**, Train, StationID, ArrivalTime, DepartureTime

Database Table: TrainSchedule

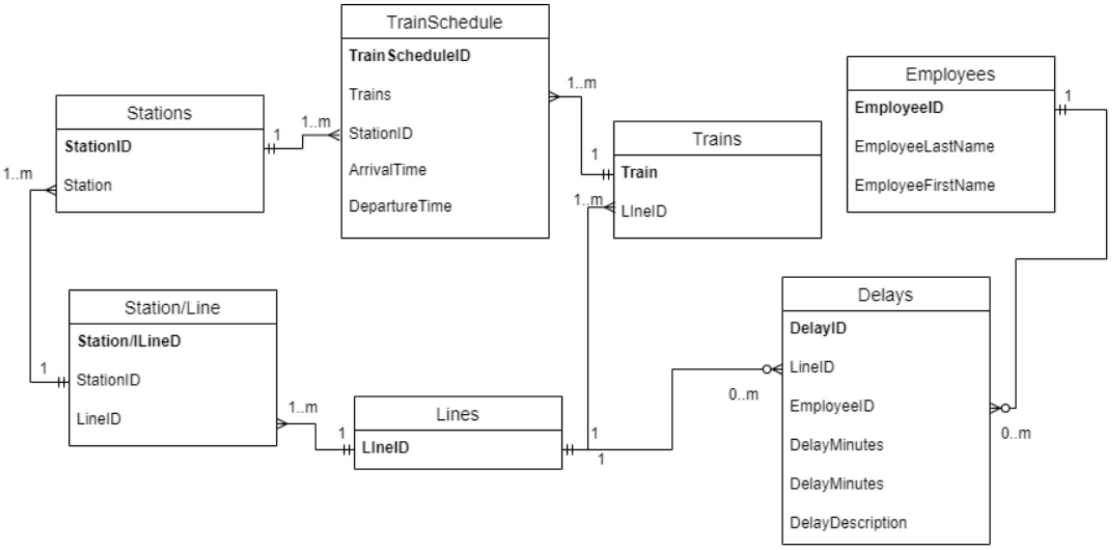


TrainSchedule

TrainS	Train	StationID	ArrivalTime	DepartureTime	(
28	R51	R100	6:15 AM	6:30 AM	
29	R51	R200	8:15 AM	8:30 AM	
30	R51	R300	10:15 AM	10:30 AM	
31	R51	R700	2:15 PM	2:30 PM	
32	R51	R800	3:15 PM	3:30 PM	
33	R52	R100	9:15 AM	9:30 AM	
34	R52	R200	10:15 AM	10:30 AM	
35	R52	R300	11:15 AM	11:30 AM	
36	R52	R700	3:15 PM	3:30 PM	
37	R53	R800	4:15 PM	4:30 PM	
38	R54	R100	12:15 AM	12:30 AM	
39	R54	R200	10:15 AM	10:30 AM	
40	R54	R300	11:15 AM	11:30 AM	
41	R54	R700	3:15 PM	3:30 PM	
42	R54	R800	4:15 PM	4:30 PM	
43	B31	B100	7:00 AM	7:15 AM	

Record: 1 of 17 No Filter Search

Database Relationship Diagram



5.4.1 Hardware:

Personal computers for programming, design, processing, and storage.
Cloud servers and other hosting services for Web and database components if implemented

5.4.2 Software:

Pycharm: Python development (Not used in final version)
Microsoft Access: Database design and management, GUI functionality

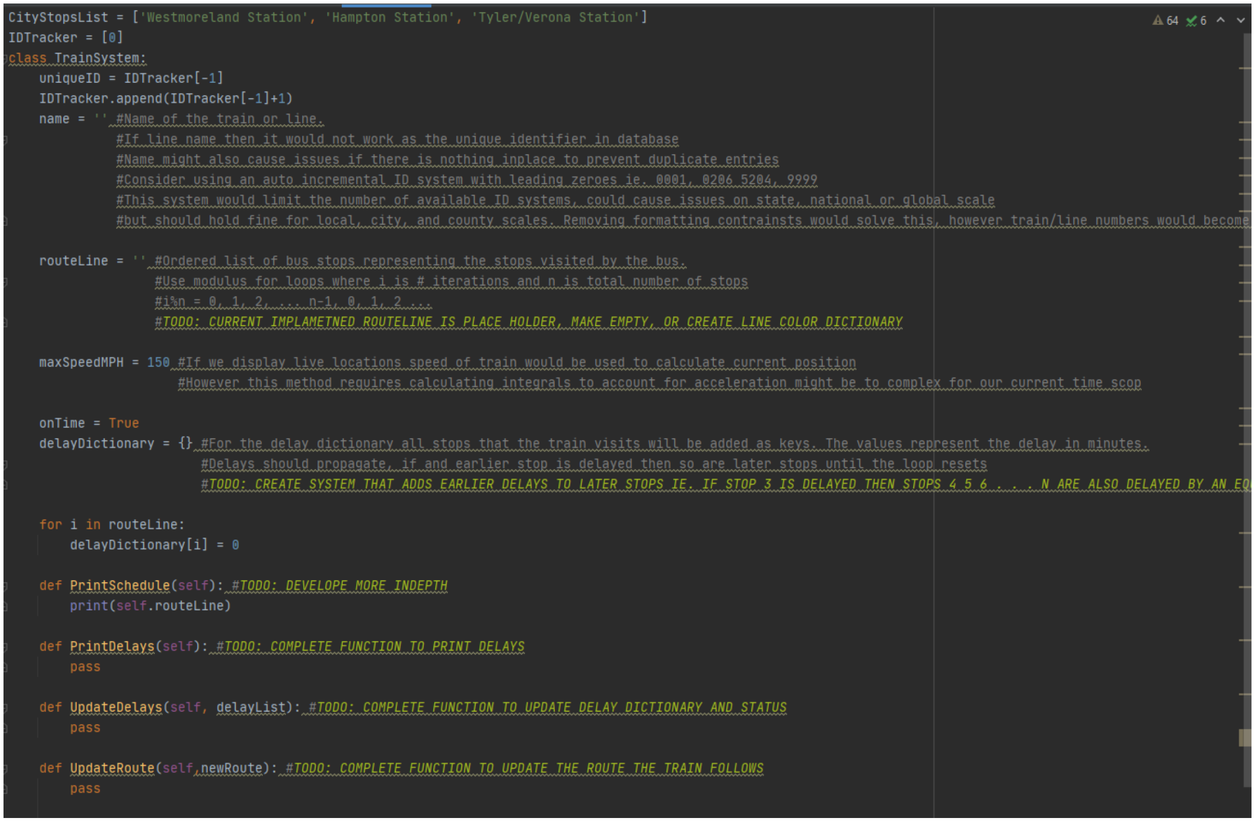
Discord: VoIP, text chat, document sharing, general communication.
Google Docs/Google Slides: Editing and creating appropriate documents for the project
MySQL(Not used in final version)

6.Implementation and Debugging

Nicolo:

Before we set about using Microsoft Access for Database, UI and computational functionality I started writing several prototypes to perform several key features such as, accessing and updating data from the main database. I also worked on a class structure for tracking and updating train information. Planned functions included: Delay simulation, train arrival and departure simulation, and unit tests to confirm that the system worked as intended. Much of this was left uncompleted in early stages as it became clear that this level of development did not fit the time scope of the project.

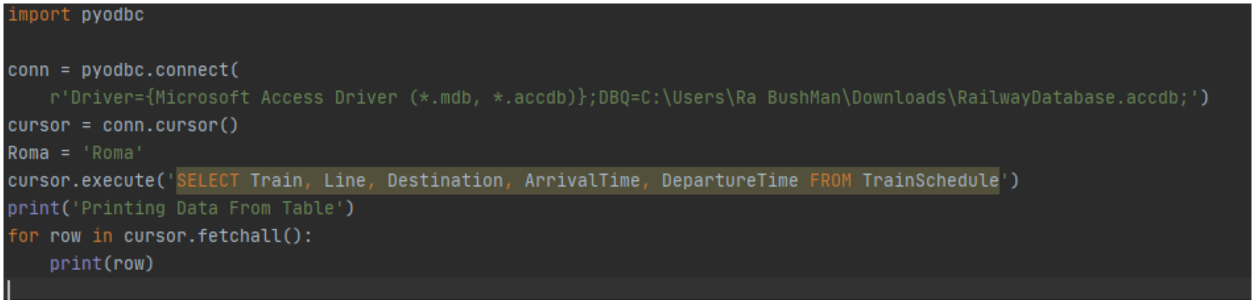
Pictured is the prototyped train class, intended for creation and simulation of train data:



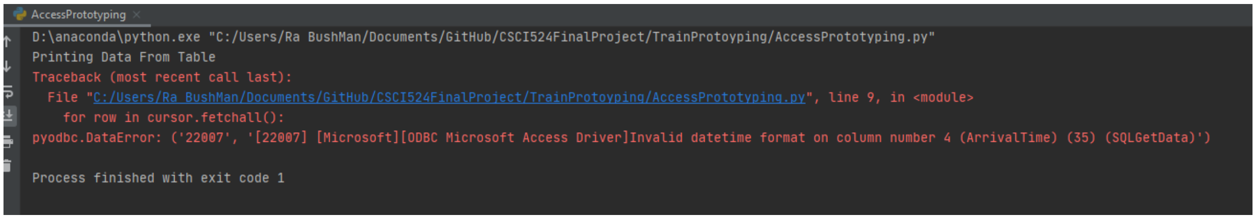
```
CityStopsList = ['Westmoreland Station', 'Hampton Station', 'Tyler/Verona Station']
IDTracker = [0]
class TrainSystem:
    uniqueID = IDTracker[-1]
    IDTracker.append(IDTracker[-1]+1)
    name = '' #Name of the train or line.
              #If line name then it would not work as the unique identifier in database
              #Name might also cause issues if there is nothing inplace to prevent duplicate entries
              #Consider using an auto incremental ID system with leading zeroes ie. 0001, 0206 5204, 9999
              #This system would limit the number of available ID systems, could cause issues on state, national or global scale
              #but should hold fine for local, city, and county scales. Removing formatting contrainsts would solve this, however train/line numbers would become

    routeLine = '' #Ordered list of bus stops representing the stops visited by the bus.
                   #Use modulus for loops where i is # iterations and n is total number of stops
                   #i%n = 0, 1, 2, ... n-1, 0, 1, 2 ...
                   #TODO: CURRENT IMPLAMETNED ROUTELINE IS PLACE HOLDER, MAKE EMPTY, OR CREATE LINE COLOR DICTIONARY

    maxSpeedMPH = 150 #If we display live locations speed of train would be used to calculate current position
                      #However this method requires calculating integrals to account for acceleration might be to complex for our current time scop

    onTime = True
    delayDictionary = {} #For the delay dictionary all stops that the train visits will be added as keys. The values represent the delay in minutes.
                         #Delays should propagate, if and earlier stop is delayed then so are later stops until the loop resets
                         #TODO: CREATE SYSTEM THAT ADDS EARLIER DELAYS TO LATER STOPS IE. IF STOP 3 IS DELAYED THEN STOPS 4 5 6 . . . N ARE ALSO DELAYED BY AN EQ

    for i in routeLine:
        delayDictionary[i] = 0

    def PrintSchedule(self): #TODO: DEVELOPE MORE INDEPTH
        print(self.routeLine)

    def PrintDelays(self): #TODO: COMPLETE FUNCTION TO PRINT DELAYS
        pass

    def UpdateDelays(self, delayList): #TODO: COMPLETE FUNCTION TO UPDATE DELAY DICTIONARY AND STATUS
        pass

    def UpdateRoute(self,newRoute): #TODO: COMPLETE FUNCTION TO UPDATE THE ROUTE THE TRAIN FOLLOWS
        pass
```

Above is the train class that I was developing for testing purposes. The train number/Unique identifier was to be generated on the creation of an instance of the class. I originally wanted a complicated speed system that would give a very realistic process of travel. However to account for acceleration, track curvature, and even rail mandated speed limits the amount of modeling and calculus required to implement that system would have taken more time than the project itself. Methods shown were intended to update the objects own parameters rather than updating the delay system, this was in line with the objective for this class which was to be used for testing our system, rather than operating the system. It became clear to me that this idea was overly complicated and would have required an analysis project in and of itself to produce a good result.

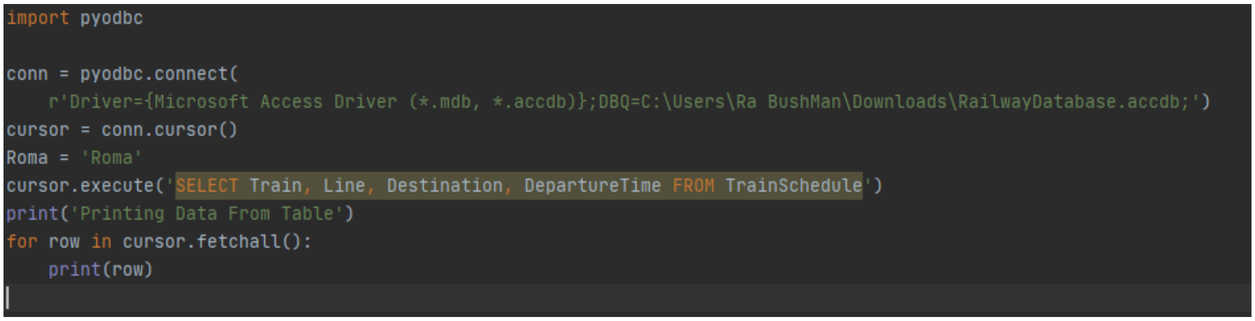
Odd data bug I ran into while trying to connect code to access database:



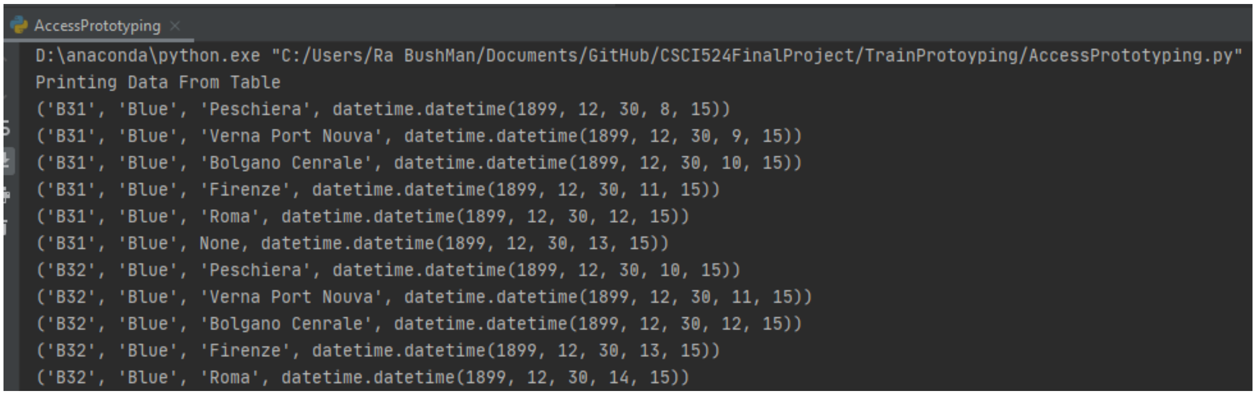
```
import pyodbc

conn = pyodbc.connect(
    r'Driver={Microsoft Access Driver (*.mdb, *.accdb)};DBQ=C:\Users\Ra BushMan\Downloads\RailwayDatabase.accdb;')
cursor = conn.cursor()
Roma = 'Roma'
cursor.execute('SELECT Train, Line, Destination, ArrivalTime, DepartureTime FROM TrainSchedule')
print('Printing Data From Table')
for row in cursor.fetchall():
    print(row)
```



```
D:\anaconda\python.exe "C:/Users/Ra BushMan/Documents/GitHub/CSCI524FinalProject/TrainProtoyping/AccessPrototyping.py"
Printing Data From Table
Traceback (most recent call last):
  File "C:/Users/Ra BushMan/Documents/GitHub/CSCI524FinalProject/TrainProtoyping/AccessPrototyping.py", line 9, in <module>
    for row in cursor.fetchall():
pyodbc.DataError: ('22007', '[22007] [Microsoft][ODBC Microsoft Access Driver]Invalid datetime format on column number 4 (ArrivalTime) (35) (SQLGetData)')

Process finished with exit code 1
```



```
import pyodbc

conn = pyodbc.connect(
    r'Driver={Microsoft Access Driver (*.mdb, *.accdb)};DBQ=C:\Users\Ra BushMan\Downloads\RailwayDatabase.accdb;')
cursor = conn.cursor()
Roma = 'Roma'
cursor.execute('SELECT Train, Line, Destination, DepartureTime FROM TrainSchedule')
print('Printing Data From Table')
for row in cursor.fetchall():
    print(row)
```



```
D:\anaconda\python.exe "C:/Users/Ra BushMan/Documents/GitHub/CSCI524FinalProject/TrainProtoyping/AccessPrototyping.py"
Printing Data From Table
('B31', 'Blue', 'Peschiera', datetime.datetime(1899, 12, 30, 8, 15))
('B31', 'Blue', 'Verna Port Nouva', datetime.datetime(1899, 12, 30, 9, 15))
('B31', 'Blue', 'Bolgano Cenrale', datetime.datetime(1899, 12, 30, 10, 15))
('B31', 'Blue', 'Firenze', datetime.datetime(1899, 12, 30, 11, 15))
('B31', 'Blue', 'Roma', datetime.datetime(1899, 12, 30, 12, 15))
('B31', 'Blue', None, datetime.datetime(1899, 12, 30, 13, 15))
('B32', 'Blue', 'Peschiera', datetime.datetime(1899, 12, 30, 10, 15))
('B32', 'Blue', 'Verna Port Nouva', datetime.datetime(1899, 12, 30, 11, 15))
('B32', 'Blue', 'Bolgano Cenrale', datetime.datetime(1899, 12, 30, 12, 15))
('B32', 'Blue', 'Firenze', datetime.datetime(1899, 12, 30, 13, 15))
('B32', 'Blue', 'Roma', datetime.datetime(1899, 12, 30, 14, 15))
```

In the above code I was trying to develop a database access code that would be used to selectively update and display data from the database. It didn't get very far as I ran into a bug that I still don't understand. In the line that starts with "cursor.exectue" You can see the SQL style query that I was using to access data to display train schedules to the public. The bug as seen in picture #2 says that the column of data called "ArrivalTimes" was not a valid format of the datetime variable. However column "DepartureTime" was the exact same format of datetime, the only difference was an offset of 15 minutes for each entry. For some reason ArrivalTime always produced this error regardless of the structure of the query while

DepartureTime never caused this error. I talked with Krystal, who was the main force behind our database, and as far as any of us could tell the columns ArrivalTime and DepartureTime were exactly the same, except for the 15 minute offset. This along with a couple other access connection bugs I couldn't workout led to the abandonment of using Python as a driving force for the system and referred to Krystal's expertise in Access to encapsulate as much as possible within Access using the built in functions.

7.Summary

Nicolo: In this project we look at the feasibility, methods and process of building a system to document, and report railway delays that are input by railway staff. The project starts with an analysis of the reason why a rail system would be needed in the first place. What the problem is, how it affects the users, and how we might try to resolve the problem. The next stage is defining the components of the problem, user and system. We look into the specifics of the problem, how it happens, how it affects people, who the people are that are affected, and who is directly and indirectly affected by the problem. We then break down our own solution. We define our requirements and develop a questionnaire to make sure our solution does not require more resources, time or effort than the current method of dealing with the problem. All the while we are prototyping and experimenting with our system to identify any flaws or mistakes in our drafted methodology, this is important to our agile work method as it lets our system experience any issues that might occur early in development. These early development issues could in future be depended upon by other sections of the system and make the fix much harder and labor intensive. By identifying them early, we can work them out before they get big.

Appendix A

A.1 Lessons learned

- Nicolo: Although prototyping and coding is how I usually visualise and get a feel for what I am trying to accomplish, when working with a group having clearly defined and laid out plans makes it much easier to keep everyone on the same page and understanding of our current stages of progress.
- Paul: Projects are always changing due to the unexpected challenges that arise but having the correct project scope/goals while using the correct project development methods, a project can still be seen through to completion.
- Krystal: Although we only made it through to iteration 3, I now begin to understand how vast a project can become. It takes very clear communication and understanding to pull a project off.
- Palash: I learned what it is like to work with a team. Everyone is good at something. With a plan and a strategy, any project can be implemented in a good shape. I learned that sometimes a task may seem to be scary but, one must not run away from their responsibilities. From the project, I learned that system analysis and design is the vital part of software development.

A.2 Division of work among team members

- Nicolo: Primarily focused on prototyping and assisting group members. Worked with Krystal and Paul to define the criteria of our system, what our goals and expectations were. Designed requirements for individual subsystems such as GUI.
- Paul: After working with the group putting together the initial project problem and what we wanted the system to accomplish, my focus was on putting our thoughts to

words in the introduction section. I was also in charge of the user stories, the use case models, and the use case descriptions.

- Krystal: My focus was on the project design and how they could be accomplished. The majority of time was spent on class domains, a sequence diagram for the system, the database and building the prototype of the project in the database.
- Palash: After studying the requirement analysis done by Paul, Krystal, and Nicolo, I designed a work breakdown structure and gantt chart that would diagrammatically explain the work done on the project. I also designed a use case diagram and sequence diagrams for the train scheduling subsystem.

A.3 Unsolved Problems

- Nicolo: In section 6 I talk about the biggest unsolved implementation problem I ran into, but there are always more problems. I think not having an automatic hands off method of testing is a big unsolved problem. Early stage manual testing is good, however the system will have to function without direct intervention or oversight so it should be tested in the same manner to erase any doubts over its functionality.
- Paul: The project was on a very tight time schedule and the group did the best they could and I was very happy with the final result. The one area I would have liked to put more time into was possibly researching more into the train scheduling boards we talked about and look at the ways that they interface with systems so we could have provided a better explanation in that area.
- Krystal: Given more time, I would like to convert my Access database into MySQL. The Access database was sufficient for the prototype but the actual application would require a more sophisticated database that could interact better with the other processes of the system.
- Palash: One unsolved problem I would like to work on is putting this system on a website which can be used by all stakeholders in real-time.

A.4 Definitions, Acronyms and Abbreviations

Customers, including but not limited to: passengers, commercial shipping parties, and civic government.

Rail employees, including but not limited to: conductors, engineers, maintenance, management, announcers, custodial employees, ticket checkers, customer service, ticket booth operators, freight and cargo loaders.

VoIP: Voice over IP, a system to facilitate vocal communication between group members.

A.5 References

LIRR - Tickets, Schedules, fares, Rail. (n.d.). Retrieved from https://www.mta.info/lirr

The Association of American Railroads (AAR). (2021, March 31). Retrieved from https://www.aar.org/

High speed Italy, Chris Smere [April 2020] from http://www.projectmapping.co.uk/Europe%20World/italytrainrailma.htm