William Widjaja
Student ID: 923651973
Github: willi301

Milestone	Date Submitted
M3	10/16/2024

Section II
Table of Contents

Section III
Project description

The database that I decided to choose is an airport management system. The reason why I chose this is because I'm an international student. I often travel back to my country during holidays without school. My home country is Indonesia and it takes around 26 hours of flight to get from the United States to Indonesia. So I have spent a lot of time in airplanes and airports. As a future software engineer, I have always wondered what I could do to make my and other's flight experience better.

Main Feature:

1. Managing flight schedule
2. Managing user ticket
3. Managing flight seat availability
4. Tracking where luggage goes
5. Airplane maintenance schedule
6. Managing airplane gate schedule

Unique Feature:

1. Can find an alternative flight when a flight is canceled
2. Can do a ticket trade with someone else
3. Automatically gives users with overnight travel give accomodation

Existing software that could benefit from this database:

- Cheapoair / google flights: they would be able to use this database to keep track of what flights are available and be able to find the price to provide customers for the cheapest flight.
- Official airports website (ex: sfo official airport): they can provide information to the customer who will be going to the airport what flights are available or having disruptions or delay. They can also give information to customer and airline companies about which terminal, gate, and check in counter for which flights are located.

Use case:

1. Use case: able to trade tickets with other user
   Actors: user1, user2
   Descriptions:
   During thanksgiving, 2 sfsu students wants to go back to their home for holidays. User1 wants to go from california to seattle to meet their family on the 15th for 200 dollars. User2 also wants to go from california to seattle to meet his girlfriend but he schedule the flight for the 19th for 100 dollars. One day, user1 suddenly gets a message from his friend saying that he has an emergency and asked for user1 for help on the 16th. So by that logic, user1 will have to miss the flight.
   With the airport management system, user1 will be able to make a post or status update saying that he needs to reschedule. User2 will see that there's a cheaper ticket

and he sees that he will be able to meet his girlfriend faster so he happily chose to trade tickets with user1.

2. Use case: automatically find alternative flight
   Actors: John
   Descriptions:

   John is going on a business trip to Italy to meet investors for his business. The meeting will be on 16th january in italy and he will be arriving to italy on the 15th because he doesn't want to spend to much time away from his business in california. On the day of his flight, he was waiting on the airport for his flight when suddenly, the gate told him that his flight is cancelled because of maintenance issue on the airplane. This will be a very big problem for John because if he's late to the meeting then he will give a bad impression to the investors.

   With this airport management system, the airport can help find another flight that is open going to italy at the same time. The price of the ticket maybe be a bit more expensive, but John doesn't really care as long as he gets to go to the meeting

3. Use case: Giving accommodations for customer with overnight travel
   Actors: Steve, United airlines, JFK (new york's airport)
   Descriptions:

   Steve works as a software engineer in san francisco so he wants to find a vacation away from the US to somewhere new. So he decides to go the farthest place he can think of, which is Indonesia. Steve decided to fly with United airlines. United airlines flight offers a flight from sfo to indonesia with a transit in Japan. However, the transit time in Japan is 22 hours so Steve will have to stay in the airport for quite along time. Steve is quite worried since he's scared of just sitting in the airport chairs for 22 hours. He fears of being to tired and missed his flight. He's also scared that when he sleeps in the chairs his belongings might get stolen

   With this Airport management system, this database will detect when a user is doing an overnight stay in the airport. United airlines notices that Steve is doing an overnight flight and they decide to contact the Narita airport to give Steve a complimentary accommodation for Steve to stay while in transit. Steve feels that his needs was served and would happily fly with united again.

Section IV
Functional Database
Entities

1. User
    a. A user shall create at most one account
    b. A user shall have a unique tracking id
    c. A user that does not have an account, doesn't have any role assigned in this system
2. Account
    a. An account shall have a unique id
    b. An account shall have a first name
    c. An account shall have a last name
    d. An account shall have date of birth
    e. An account shall have a passport id
    f. An account shall have many ticket
    g. An account shall have many luggage
3. Ticket
    a. A ticket shall have at most one flight
    b. A ticket shall have at most one Date
    c. A ticket shall have belong to at most one Account
    d. A ticket shall have at most one seat
    e. A ticket shall have at most one Gate
    f. A ticket shall have a unique id
    g. A ticket shall have an overnight status
    h. A ticket shall have a price
4. Plane
    a. A plane shall have many seat
    b. A plane shall have a status (in need of maintenance, in maintenance progress, available, in use)
    c. A plane shall have at most one airline company
    d. A plane shall have only one airplane type
    e. A plane shall have at most one flight
5. Flight
    a. A flight shall have a plane
    b. A flight shall have at most one departure date
    c. A flight shall have at most one arrival date
    d. A flight shall have a destination airport
    e. A flight shall have a departure airport
    f. A flight shall have a status (delayed, on time, cancelled)
    g. A flight shall have at most one gate
6. Date
    a. A date shall have at most one year
    b. A date shall have at most one month
    c. A date shall have at most one date

    d. A date shall have at most one hour
    e. A date shall have at most one minute
7. Gate
    a. A gate shall have a gate number
    b. A gate shall have a location
    c. A gate shall have at most one Plane
    d. A gate shall be own by only one terminal
    e. A gate shall have many airline company
8. Luggage
    a. A luggage shall belong to only one Account
    b. A luggage shall be connected to only one plane
    c. A luggage shall be specified whether it's fragile / oversized ?
9. Pilot
    a. A pilot shall have one plane
    b. A pilot shall have a unique id
    c. A pilot shall have a name
    d. A pilot shall have a status (active, standby, off duty)
    e. A pilot shall have a airline company
    f. A pilot can also be part of flight crew (recursion)
10. Seat
    a. A seat shall have a type (economy, business, first class, etc)
    b. A seat shall have at most one ticket
    c. A seat shall be own by one plane
    d. A seat shall have a number
11. Ground staff
    a. A Ground staff shall have at most one airline company
    b. A Ground staff shall have a role (ticketing, maintenance, transportation, etc)
    c. A Ground staff shall have a status (active, standby, off duty)
12. Terminal
    a. A terminal shall have a unique id
    b. A terminal shall have an amount of gate available to use
    c. A terminal shall have an amount of gates in use
13. Airline company
    a. An airline company shall have a unique id
    b. An airline company shall have a name
    c. An airline company shall have many gates
    d. An airline company shall have a number of how many plane on ground
    e. An airline company shall have a number of how many plane on air
14. Check-in counter
    a. A check-in counter shall have a gate
    b. A check-in counter shall have a status (open or closed)
    c. A check-in counter shall have an airline company
15. Parking
    a. A parking shall have one terminal

    b. A parking shall have a code (e.g: a1, b10)
    c. A parking shall have a type (long term, over night, short term)
16. Airport Hotel
    a. An airport hotel shall have one terminal
    b. An airport hotel shall have a location
    c. An airport hotel shall have a rate
    d. An airport hotel shall have a type (budget, business, luxury)
    e. An airport hotel shall have a number of available room
17. Flight crew
    a. A flight crew shall have a name
    b. A flight crew shall have a unique id
    c. A flight crew shall have a status (on duty, available, off duty)
    d. A flight crew shall have one airline company
    e. A flight crew shall have a flight
18. Baggage claim
    a. A baggage claim shall have a unique id
    b. A baggage claim shall have a luggage
    c. A baggage claim shall have a status (claimed, not claimed, lost)
19. Feedback
    a. A feedback shall have a unique id
    b. A feedback shall have a type (hygiene, tech, personal problem, etc)
    c. A feedback shall have location
20. Lost-and-found
    a. A lost-and-found shall have a unique id
    b. A lost-and-found shall have a type (baggage, handbag, electronic)
    c. A lost-and-found shall have a status (claimed, unclaimed, etc)

Section V

Performance

- This database should be able to be accessed on multiple devices at the same time
- The database should be accessed and processed in a fast time
- A protocol should be established so that the database will not crash at anytime

Security

- The database should two-factor authentication so only authorized people should be allowed to have access to the database
- The database should have a secure way for user to input their personal information to the database
- The database should be encrypted so people wouldn't be able to see the data easily.

Scalability

- The database should have resources according to the size to the data that the database holds
- The database should show the data in a presentable manner no matter the size of the database.
- The database should be able to hold as much data as it is given

Capability

- The database should be capable of working in all Operating system
- The database should also work on any kind of devices
- The database should be capable of updating to the appropriate version when a new version is released. When there is an issue with the current version, then it can go back to a previous version in order to keep the database running

Environmental

- The database should be able to provide data digitally so it will reduce the usage of paper
- The database should be run efficiently to reduce electricity usage
- The database should encourage the usage of e-tickets

Coding standards

- All variables' name should be written in camel case standard
- All coding should make sure with the appropriate tab for better readability
- Dividing code into code blocks with the appropriate comments describing what each code blocks do

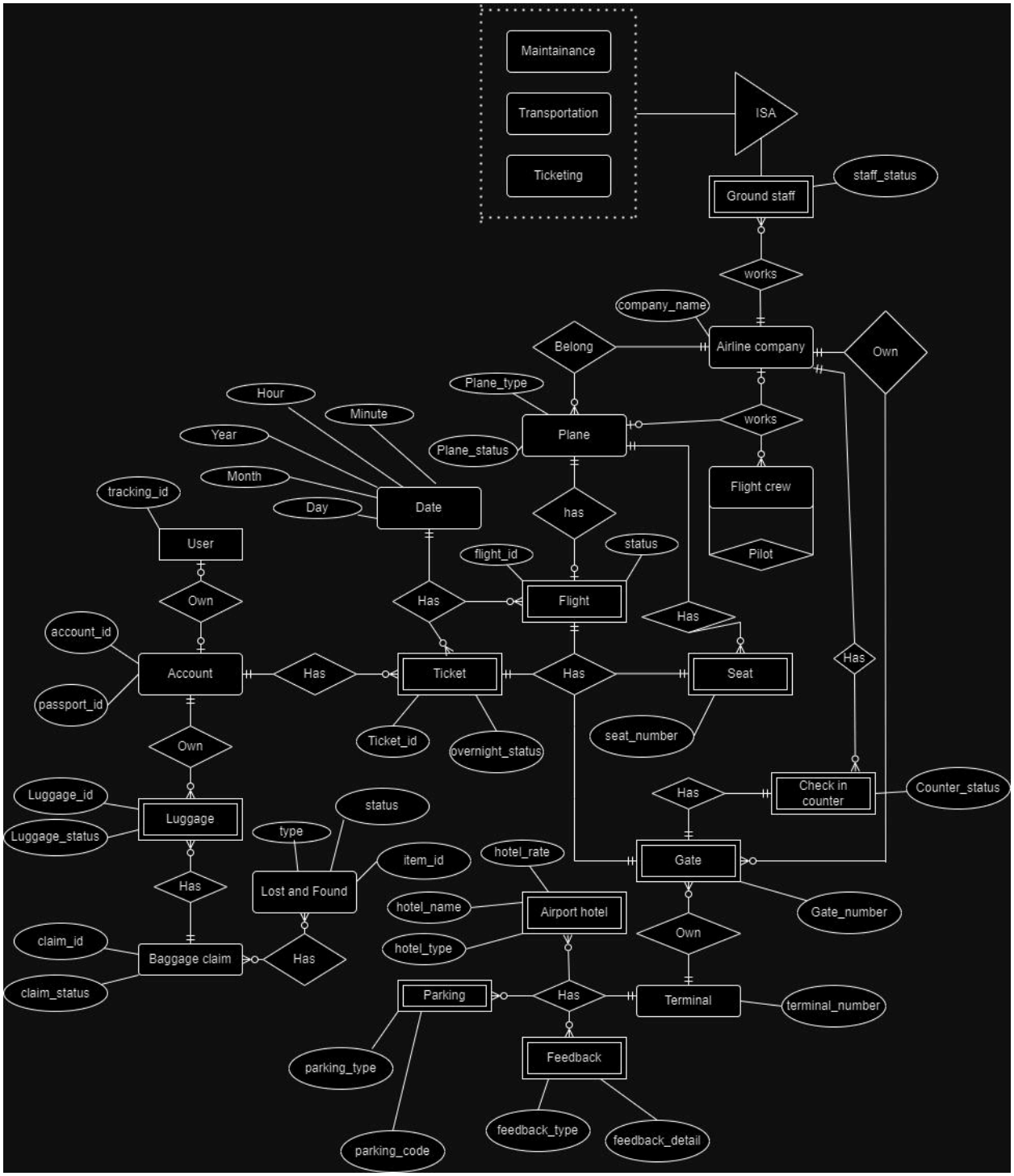
Storage

- Applications and devices should only store data that is relevant to the present time
- Devices will only store data from the past when it is required to save storage
- The database should have a backup of the data in a cloud or other medium to avoid any problem when there’s a tech issue

Privacy

- Private information such as passport number, and user ID, should only be allowed to be accessed by authorized staff
- A password system shall be created to avoid public accessing the database
- The program should encrypt the data input of a user when they input it to the database

Section VI: Entity Relationship Diagram (ERD)



Section VII: Entity Set Description

1. User (Strong)
    a. Tracking_id: strong, numeric

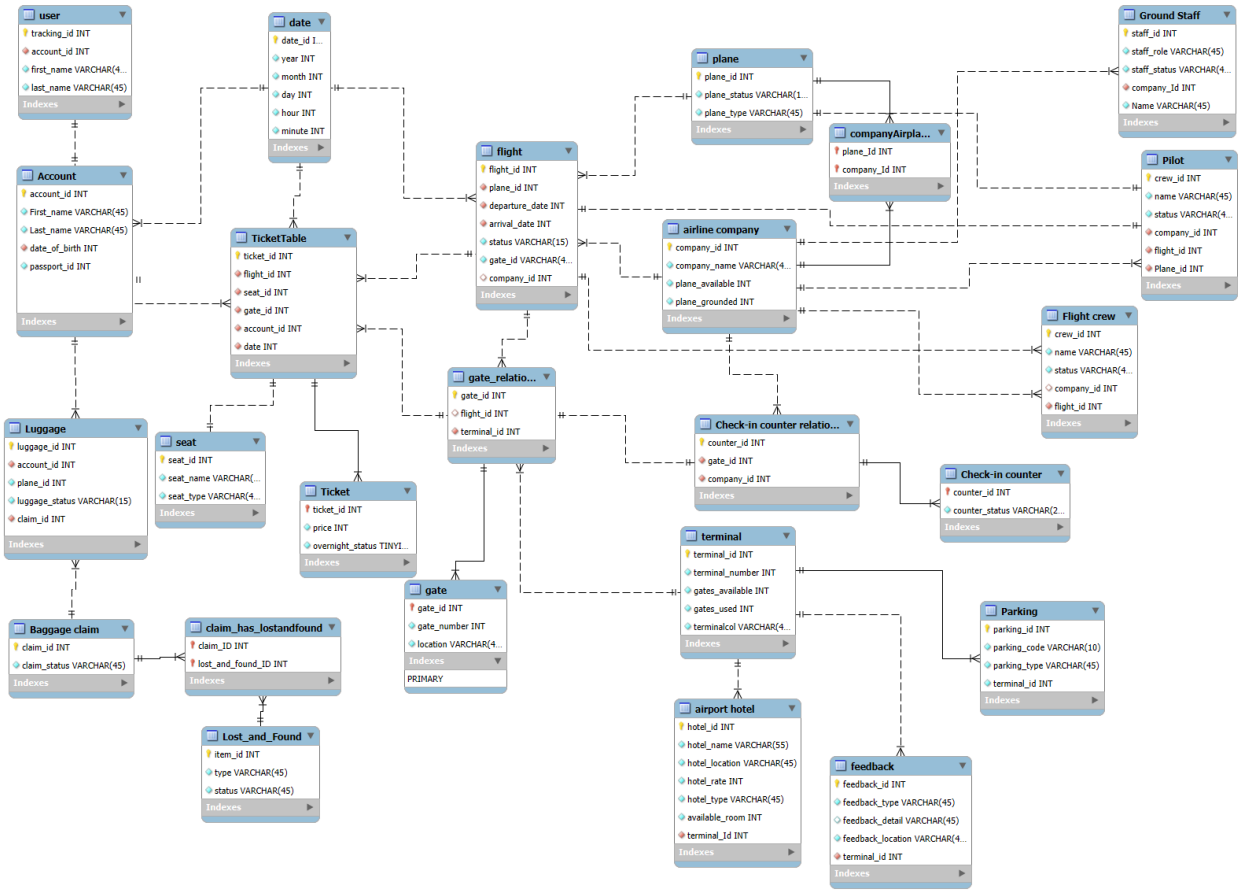
    b. account_id: composite primary key foreign key, numeric, (references account entity set)
    c. First_name: alphanumerical
    d. Last_name: alphanumerical
2. Account (Strong)
    a. account_id: composite primary key foreign key, numeric
    b. First_name: alphanumerical
    c. Last_name: alphanumerical
    d. Date_of_birth: Date
    e. Passport_id: numerical
    f. Ticket_id: numeric, composite primary key foreign key (references to ticket entity set)
    g. Luggage_id: numeric, composite primary key foreign key (references to luggage entity set)

3. Ticket (weak)
    a. Price: numeric
    b. Ticket_id : numeric, composite primary key foreign key
    c. Account_id: numeric, composite primary key foreign key (reference to account entity set)
    d. Overnight_status: boolean
    e. Flight_id: numeric, composite primary key foreign key (references flight entity set)
    f. Seat_id: numeric composite primary key foreign key (reference seat entity set)
    g. Gate_id: numeric composite primary key foreign key (reference gate entity set)
4. Flight (weak)
    a. Flight_id: numeric, primary key
    b. plane_Id: numeric, composite primary key foreign key (references plane entity set)
    c. Departure_date: composite primary key foreign key(reference date entity set)
    d. Arrival_date: composite primary key foreign key(reference date entity set)
    e. Status: alphanumerical (delayed, on-time, cancelled)
    f. Gate_id: numerical, composite primary key foreign key (reference gate entity set)
5. Plane (Strong)
    a. Plane_id: numeric, primary key
    b. Plane_status: alphanumeric, (need maintenance, in use, available, in maintenance)
    c. Company_id: numerical, composite primary key foreign key (reference to airline company entity set)
    d. Plane_type: alphanumeric
    e. Flight_id: numeric, composite primary key foreign key (reference to flight entity set)
6. Gate (weak)
    a. Gate_number: numerical
    b. Gate_id: numerical, primary key
    c. Location: alphanumerical
    d. Plane_id: numeric, composite primary key foreign key (reference to plane entity set)
    e. terminal _id: numeric, composite primary key foreign key (reference to terminal entity set)

    f. Company_id: numerical, composite primary key foreign key (reference to airline company entity set)
7. Luggage (weak)
    a. Luggage_id: numeric, primary key
    b. Account_id: numeric, composite primary key foreign key (reference to account entity set)
    c. plane_Id: numeric, composite primary key foreign key (references plane entity set)
    d. Luggage_status: alphanumeric (fragile, oversized, normal)
8. Airline company (strong)
    a. company_Id: numerical
    b. Company_name:alphanumerical
    c. Plane_available:numerical
    d. plane_grounded:numerical
9. Flight crew (Strong)
    a. Name: alphanumerical
    b. Crew_id: numerical, primary key
    c. Status: alphanumerical (on duty, available, off duty)
    d. Company_id: numerical, composite primary key foreign key (reference to airline company entity set)
    e. Flight_id: numeric, composite primary key foreign key (reference to flight entity set)
10. Seat (weak)
    a. Seat_number: alphanumerical
    b. Seat_id: numerical, primary key
    c. Seat_type: alphanumerical, (economy, business, first class)
    d. Ticket_id: numeric, composite primary key foreign key (references to ticket entity set)
11. Terminal (Strong)
    a. Terminal_number: numerical
    b. Terminal_id:numeric, primary key
    c. Gates_available: numerical
    d. Gates_used: numerical
12. Parking (weak)
    a. Parking_code: alphanumerical, primary key
    b. terminal _id: numeric, composite primary key foreign key (reference to terminal entity set)
    c. Parking_type: alphanumerical, (overnight, short term, long term)
13. Airport hotel (weak)
    a. terminal _id: numeric, composite primary key foreign key (reference to terminal entity set)
    b. Hotel_name: alphanumeric
    c. Hotel_location: alphanumeric
    d. Hotel_rate: numeric
    e. Hotel_type: alphanumeric (budget, business, luxury)
    f. Available_room: numeric
14. Ground staff (weak)

    a. Company_id: numerical, composite primary key foreign key (reference to airline company entity set)
    b. Staff_role: alphanumeric (maintanance, ticketing, transportation)
    c. Staff_status: alphanumeric (off duty, standby, on duty)
15. Baggage claim (Strong)
    a. Claim_id: alphanumeric, primary key
    b. Luggage_id: numeric, composite primary key foreign key (reference luggage entity set)
    c. Claim_status: alphanumeric (claimed, not claimed, lost)
16. Date (strong)
    a. Date_id: numeric, primary key
    b. Year: number
    c. Month: number
    d. Day: number
    e. Hour: number
    f. Minute: number
17. Feedback (weak)
    a. Feedback_id: numeric, primary key
    b. Feedback_type: alphanumeric (hygiene, tech, personal problem, etc)
    c. Feedback_detail: alphanumeric
    d. Feedback_location: alphanumeric
    e. terminal _id: numeric, composite primary key foreign key (reference to terminal entity set)
18. Pilot (recursion)
    a. Name: alphanumerical
    b. Crew_id: numerical
    c. Status: alphanumerical (on duty, available, off duty)
    d. Company_id: numerical, composite primary key foreign key (reference to airline company entity set)
    e. Flight_id: numeric, composite primary key foreign key (reference to flight entity set)
    f. Plane_id: numeric, composite primary key foreign key(reference to plane entity set)
19. Lost and found (weak)
    a. Item_id: numeric, primary key
    b. Type: alphanumeric (baggage, handbag, electronic)
    c. Status: alphanumeric (claimed, unclaimed, etc)
    d. Claim_id: alphanumeric, primary key
20. Check in counter (weak)
    a. Gate_id: numeric composite primary key foreign key (reference gate entity set)
    b. Counter_status: alphanumeric, (open, closed)
    c. Company_id: numerical, composite primary key foreign key (reference to airline company entity set)

Section VIII: Enhanced Entity-Relationship (EER) Diagram (EER)



Section IX: Normalization Techniques Used

- In the ERD, the ticket entity has a lot of non-key entities but also have a lot of key that connects the ticket entity to other entity. For normalization purpose, i decide to create another entity called the ticketTable where all the key that is connected to the previous table will be put there. All the non-key attributes will be put in another ticket table and connect it to ticketTable
- I have also done this to the plane and companies. Previously there is a foreign key in plane that shows which company own this certain plane. However, i added a companyOwned table that holds all the key that connects plane to company