Team 11 Project Binder

Meeting 0: online via discord

Agendas:

- **Team introduction**
- **Prepare questions for interview on 2nd February**
- **Plan First Meeting**
- **Discuss team roles**

Key Points of Discussion- Team Introduction

- Since we are just starting to know each for the first time as we were unfamiliar with ourselves we started off by briefly sharing each team member's name and a little bit about their background and experience

Key Points of Discussion- Prepare questions for interview on 2nd February

- We spent several hours preparing for a customer interview on Discord that's coming up on 2nd February. We began by reading through the Airvia spec in detail, taking notes and trying to gain a clear understanding of the product and its target users.

Key Points of Discussion- Plan First Meeting

- We started to plan our first team in-person meeting, which we've scheduled to take place next week on the 6th of February. We've been communicating primarily through Discord messages up until now, where we conversed with each team member to confirm their availability and preferences for the date, time, and location. We were able to find a time that works for everyone, and we've chosen a location that's easily accessible for all team members.

Attendees:

The entire team attended as it was discord conversion

Meeting 1: 6th February, Time: 1pm - 2pm

Agendas:

- **Choose Team Name and Team Logo**
- **Picked 2 roles for each member (Primary and Secondary)**
- **Setting up appointments with consultant**
- **Decided what Programming Language and UML diagramming software**

Key Points of Meeting-Team name and logo

-Team names and logos discussed

-First suggestion was from Khalid:

-Also Code Commanders and Tech Titans was suggested by hattice

-However we decided to go as a team as Alt+F4 which was Tony Suggestion

Key Points of Meeting- Chose a Programming Language and UML diagramming software
-Chosen program for UML diagramming software: Visual Paradigm
-Chosen programming language: Java

Key Points of Meeting- Picked 2 roles for each member (Primary and Secondary):

Manager: Khalid
Deputy: Hatidzhe
System Analyst - **primary:** Aronica and Nasim
secondary: Hatidzhe
System Designer - **primary:** Methusha
secondary: Joel
Programmer - **primary:** Joel
secondary: Tony and Aronica
System Tester - **primary:** Tony
secondary: Khalid and Methusha

Key Points of Meeting- Setting up appointments with consultant
- Planned an online consultant meeting via teams for the 9th of February.

Attendees:

The entire team except nasim attended the team meeting

Customer Interview: 02/02/2023:

Customer Interview Questions:

- Is the new system supposed to produce reports in the same format as in the doc? (These Question were asked by Joel)
    - **A.** no, but the data needs to be preserved
    - Could a more clear list of data or sketch of a new template for UI be provided in the next interview?
        - **A.** no
- Clarification for reports (These Question were asked by Khalid)
    - "Global"
        - **A.** indicates the summary report showing totals of sales by advisors in agency
    - "Non-assess amount"

        - **A.** The remainder after commission
    - “Commision”
        - **A.** How much an agency keeps when they resell an AirVia ticket
    - Dash line in “assigned blanks”
        - **A.** Indicates the range of ticket numbers assigned
    - Agent “code” in ticket report
        - **A.** Indicates advisor ID

- Payment types (These Question were asked by Aronica)
    - **A.** Combination of cash and credit card is allowed
- There are taxes in the reports, is the system supposed to calculate that or is that the job of the advisors or someone else to input into the GUI? (These Question were asked by Joel)
    - **A.** The system should calculate taxes and there should be a feature where managers can configure tax % for the Agency in the GUI
- If a pay later ticket is sold, should the value of the ticket be counted to the sales report before the customer has paid for it? (These Question were asked by Joel)
    - **A.** Yes, it should be counted to final report, but there also should be an addition to the report of what fraction is yet to be paid
- What do you have in mind for a backup facility? (These Question were asked by Joel)
    - **A.** Not sure what system, they expect us to research this and make a recommendation
    - **A.** Expect back up to be both automatic, with a modifiable frequency and also to have a manual “backup now” option
- One of the requirements is a clear and consistent GUI, do you have any examples or expectations on the style or implementation of it? (These Question were asked by Khalid)
    - **A.** No specific expectations, as long as it is intuitive and looks aesthetically pleasing
- Explain how the commission rate works for refunds (This question was asked by aronica)
    - **A.** When a ticket is refunded the commission amount by the agency is refunded as well

The rest of the interview questions were asked by Joel

- Is there any significance in the difference in digits for the blanks number eg it says 6-8 numbers after the initial coupon code
    - **A.** 6 digits are for domestic sales, 8 for international.

- Have you established any specific performance requirements for the software such as response time, memory usage, and reliability?
    - **A.** No strict requirements or value, however suggested it to be reasonable such as no longer than a 1-2 sec 90% of the time.
- Can you provide a test environment for the software to be tested on, including hardware specifications, operating system, and network configurations?
    - **A.** For this prototype task, for now no requirements - may be requirements later if we “get recruited for further development”. As long as it runs on our laptop on the demo
- What is the desired user experience for the travel agents and customers using the software to report and analyse sales data?
    - **A.** Purpose is for this to be a tool that enables an agency to report back on how many AirVia tickets are sold and with how much commision
    - **A.** Prototype only needs to work for one agency

Review of Customer Interview

Given the fact the team wasn’t organised and haven’t met, half the team attended the customer interviews and were able to come up with good questions via Discord in a relatively short time.
we covered those that they hoped we would ask - the Customer (Dr Stankovic) will forward our good performance to Martin (Module Leader)
Don’t take everything he said as set in stone for details, as he is only representing Martin. Martin has the final say.
For further clarification, ask Martin.

Attendees:

Only Joel, Aronica and Khalid Attended the Customer Interview

Team Deadline

Last Task of Khalid as project Manager was to organise the deadline of the group
Which was:
Current team Deadline For First deadline on Week 5:
10 key Use cases diagram (1st draft) - Tuesday 14 Feb

Use case specification and prioritisation finalise ver. including detailed interactions with actors -Thursday 17 Feb
Design class diagram (1st draft) and ER (1st draft) diagram -Tuesday 21 Feb
Entity relationship diagram (2nd draft) and finalise class diagram -Friday 24 Feb
GUI design and finalise entity relationship diagram -Tuesday 28 Feb
Improve and finish everything to hand in document -Friday 3 March

1st Consultant Meeting : 9th February Time: 12.30am to 1pm

- Discuss team roles and progress throughout the first week
- Asked questions about how we the team members are going to be accessed
- Also Discussed why the project manager didn't email the consultant due to roles not being finalised
- Discussed when the group and consultant can meet again planned for a Tuesday 2pm meeting however cancelled due to a university strike.

Attendees:

The entire team attended the consultant meeting via zoom

Meeting 2: 13th February Time: 12.15am to 1pm

Agendas:

- Khalid and hattice switched roles

- Nasim unresponsiveness in the team project

- Planning a regular consultant meeting

Key Points of meeting- Khalid and hattice switched roles
-Khalid decided to switch to deputy allowing hattice to take reins as she has more experience in project management as was more familiar with agile software
-There was a lack of direction in terms of the sub-tasks and the team needed to act fast to meet the tight deadline.
-Also Khalid was more familiar with the role of a deputy manager and was maintaining the project binder even though he should of finished the summary of the existing system

Key Points of meeting- Nasim unresponsiveness
-we discussed a team member being slightly unresponsive
-Nasim wasn't directly communicating with the other system analyst
-He was sending his use case draft to the project manager and not directly to the group
-we decided that we are very close in taking a decision on him if he doesn't participate
-Even if he couldn't attend the meeting, he could at least be active in the discord chats.

Key Points of meeting- other Matters
-discussing when we could meet the team consultant as we haven't set a regular time in order to meet the project manager
-some team members were disgruntled that the planned meeting was cancelled due to the strike.
-Khalid told the New project Manager that he shouldn't have managed the project binder and that was a miscommunication and that now are roles are swapped khalid should continue maintaining the project binder and Hattice needs to write a summary based on the Airvia system and as she build rapport with the team consultant already we both agreed that switching our roles greatly benefits us and the team.
-The project manager (Hattice) and the system manager decided to work on the first draft nasim sent as he isn't contributing a lot to the task and that we need to complete a use case diagram in order for the system designer and programmer to continue with their tasks.

So the new roles of the Team project was finalised:

Final Roles:

Manager: Hatidzhe
Deputy: Khalid
System Analyst - **primary:** Aronica and Nasim
secondary: Hatidzhe
System Designer - **primary:** Methusha
secondary: Joel

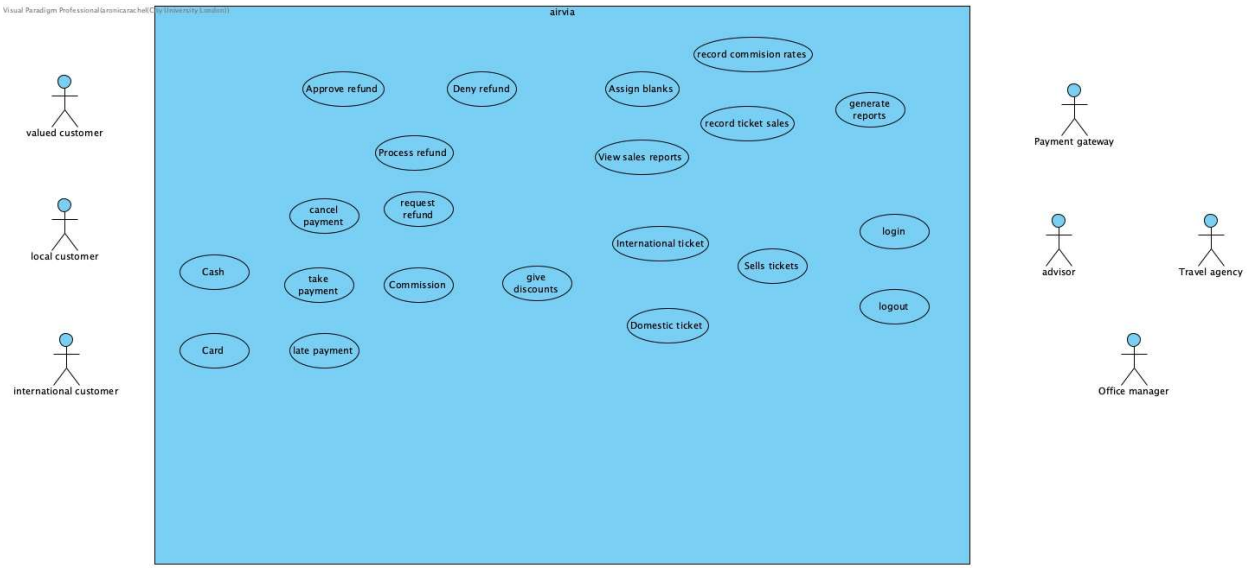
Programmer - **primary:** Joel
secondary: Tony and Aronica
System Tester - **primary:** Tony
secondary: Khalid and Methusha

Attendees:

The entire team attended the meeting except Nasim.

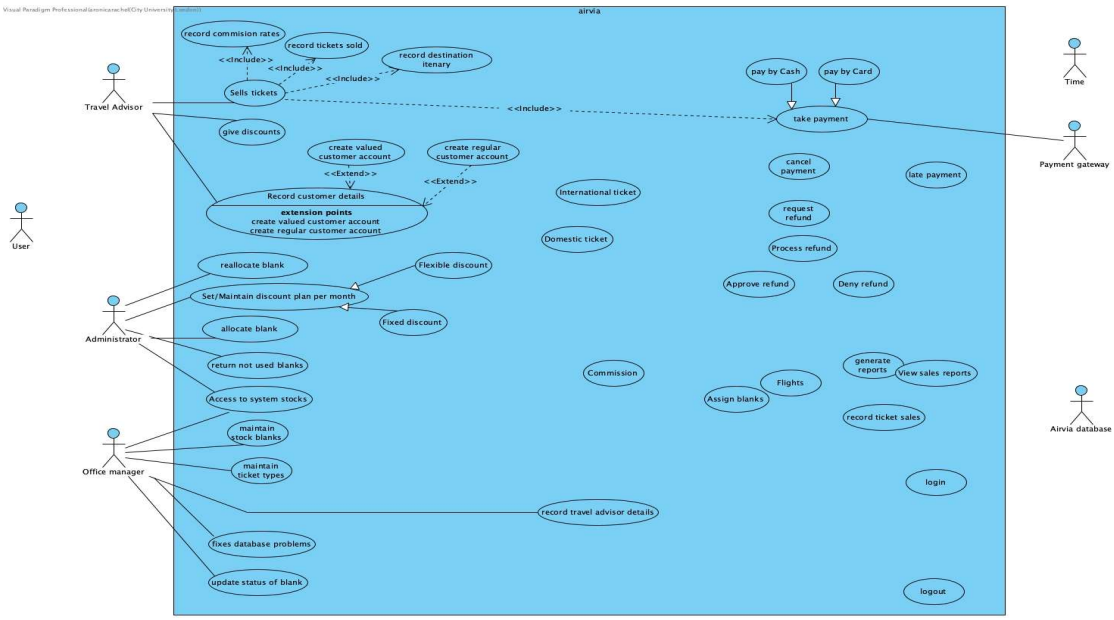
Use case drafts:

First Draft of use case diagram 15/02/2023 :



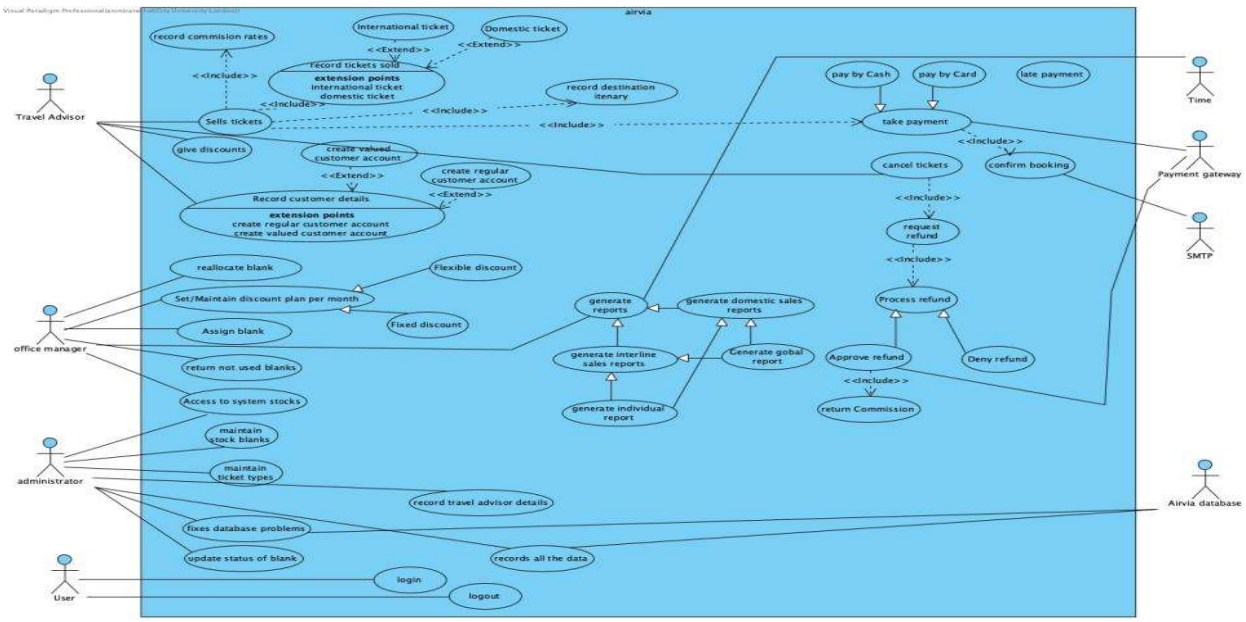
This draft was worked on by Aronica, Hattice and Khalid.

Second Draft of use case diagram 16/02/2023 :



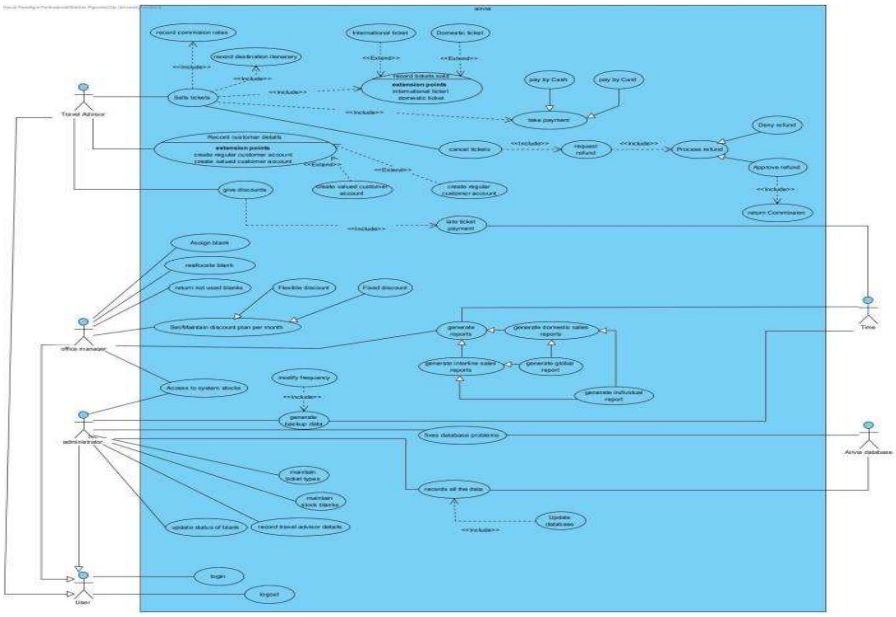
This draft was worked on by Aronica, Hattice and Khalid.

Third Draft of use case diagram 18/02/2023 :



This draft was worked on by Aronica and Hattice.

Fifth Draft of use case diagram 02/03/2023 :



This draft was worked on by Aronica.

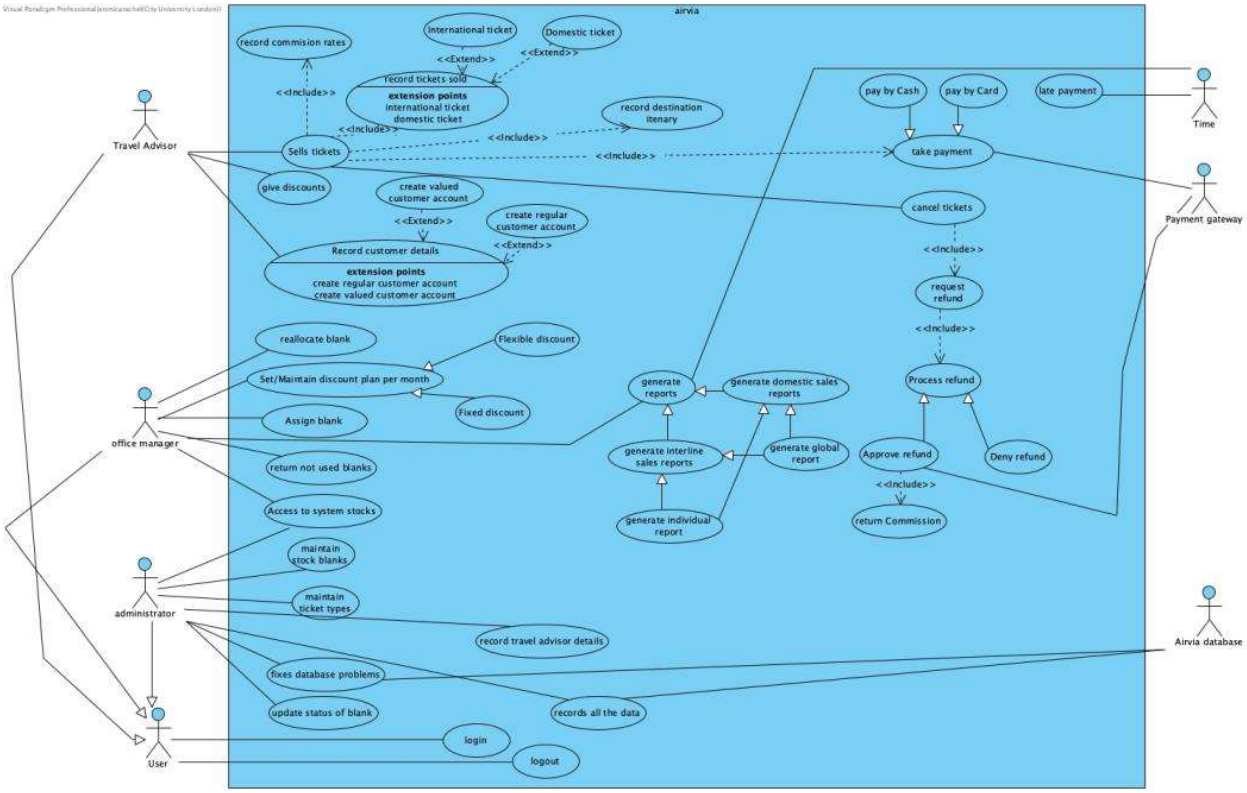
Meeting 3: 20th February Time: 12am to 12.50am

Agendas:

- **Discussion of the use case diagram**
- **Setting our github repository**
- **Planning for the next stage of our coursework**

Key Points of meeting- Discussion of the use case diagram
- The entire group contributed in assisting our use case diagram as our use cases are fundamental in creating a database and to implement design so the group stated the diagram to be solid.Also check if any parts need to be changed or improved

Fourth Draft of use case diagram 20/02/2023 :



This draft was worked on by the entire team.

Key Points of meeting- Setting our github repository

- As our project started to develop we needed a perfect place to organise our workload so Khalid opened a github repo and added all the team members to access their group work. As the secondary tester, khalid was put in charge of final checks and commits to the github to double check that nothing incorrect is added to the project.

Key Points of meeting- Planning for the next stage of our coursework

- the group is starting to become more collaborative as methusha and joel are starting to implement design given with the analysis we already have. Whilst hattice and Aronica are working on the use case specifications and to finalise the use case diagram, Khalid and Tony discuss the creation of test cases to be done after the tasks of the analysis and design.

- The team was therefore organised into sub-groups: the Analysis team, the Design team and the tester & QA team.

Attendees:

The entire team attended the meeting.

2nd Consultant Meeting : 20th February Time: 1pm to 2pm

Agendas:

- **Meet in person with consultant**
- **Introduce ourselves and the current team progress**

Key points in the meeting- Meet in person with consultant
-The team had the opportunity to meet in person with a team consultant.Being able to interact face-to-face allowed for a deeper level of understanding and a more personal connection.
-The open and candid communication we had has given the team a renewed sense of motivation and inspiration as to the fact that we would be supported if we would need further support.
Key points in the meeting- Introduce ourselves and the current team progress
- During our meeting with the consultant, we took the opportunity to introduce ourselves and our roles within the project team.
-We also provided an update on our current progress, highlighting the successes and challenges we've faced so far.
-This helped the consultant gain a better understanding of our team's dynamic and where we currently stand in the project, allowing for more targeted and effective guidance and support.

Attendees:

The entire team attended the meeting.

Meeting 4: 20th February Time: 1pm to 2pm

Agendas:

-discuss what else to add to our case diagram
-planned questions to ask Mr lancaster

Key points in the meeting- discuss what else to add to our case diagram
- After the consultant meeting we decided to add further ideas to our analysis team for them to take away so they could finalise a final draft for the use case diagram

Key points in the meeting- planned questions to ask Mr lancaster
- The team had further questions that needed further clarification from the client mr lancaster. Our Programmer had a question as to where the code of our backup of the Airvia database would be stored.
- also our project manager had a concern whether we would use a wireframe in visual paradigm.

Attendees:

The entire team attended the meeting except methusha

Meeting 5: 27th February Time: 12am to 1pm

Agendas:

- confirming who worked on which draft
- discussed key questions needed to be answered by module leader

Key points in the meeting- confirming who worked on which draft
-The first talking point of the meeting was discussing who contributed in the design stage and for the 4 drafts of design class diagrams was worked on by Methusha and Joel and for the 2 ER diagrams it was worked on by joel.

Key points in the meeting- discussed key questions needed to be answered by module leader

The team discussed some necessary key points that need clarifications by module leader:

- Where does the backup data go?
- Do we design wireframes on visual paradigm or on any other platforms?
- How long is the customer report and what is its format?

Attendees:

The entire team attended the meeting except Tony

3rd Consultant Meeting : 27th February Time: 1pm to 1.30pm

Agendas:

- **Discussed concerns about the module leader**
- **Discussed the current team progress**

Key points in the meeting- Discussed the current team progress

- The team discussed to the consultant the current progress they have made so far and highlighted the key points from the meeting. They discussed the progress on each task, identified important issues, and outlined the plan for the next steps. Everyone agreed that the team is making good progress and is committed to achieving their goals

Key points in the meeting- Discussed concerns about the module leader

- The team discussed the issue of the module leader taking too long to respond to emails, which was slowing the team's progress on the project.

Attendees:

The entire team attended the meeting except Tony

Meeting 6: 27th February Time: 1.30pm to 2pm

Agendas:

- **went over the SQL draft**
- **discussed the ER diagram**
- **discussed the class diagram**

Key points in the meeting- went over the SQL draft

- During the team meeting, the programmer (Joel) went over the SQL draft and explained his progress in the code. The deputy manager (Khalid) advised the programmer to test his code on MySQL as certain tables had typos and had incorrect syntax.The programmer also mentioned that he was still working on the SQL queries, but was waiting for a response from the module leader to clarify a question he had. It was agreed that the programmer should continue working on the SQL queries and test the code before submitting it for review.

Key points in the meeting- discussed the ER diagram

- During the team meeting, the programmer presented the Entity Relationship diagram and discussed the first draft. After he explained the relationship between the tables, the team was able to give their opinions and ideas on the diagram. With the team's help, the programmer was able to finalise the Entity Relationship diagram and make necessary adjustments to it.

Key points in the meeting- discussed the class diagram

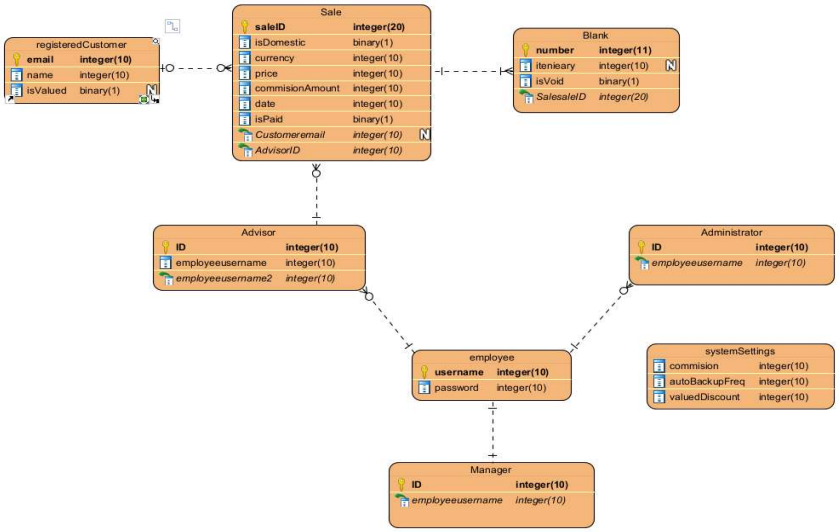
- During the team meeting, the programmer and system designer presented their three drafts of the design class diagram since the last meeting. The team discussed the drafts and made an extra draft, although the diagram was not yet finalised. It still needed to be fixed with links between the classes and more classes had to be added in order for the GUI design stage of the project to start as soon as possible.

Attendees:

The entire team attended the meeting except Tony

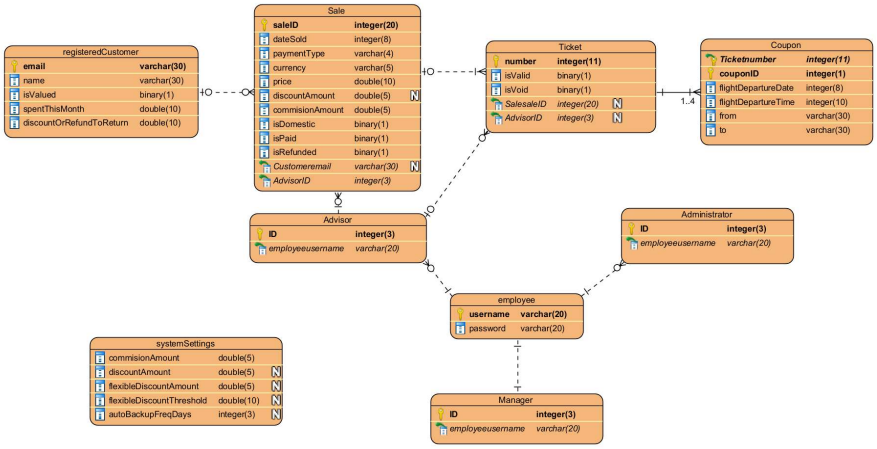
Entity-Relationship Diagrams

First Draft of Entity-Relationship Diagram 25/02/2023 :



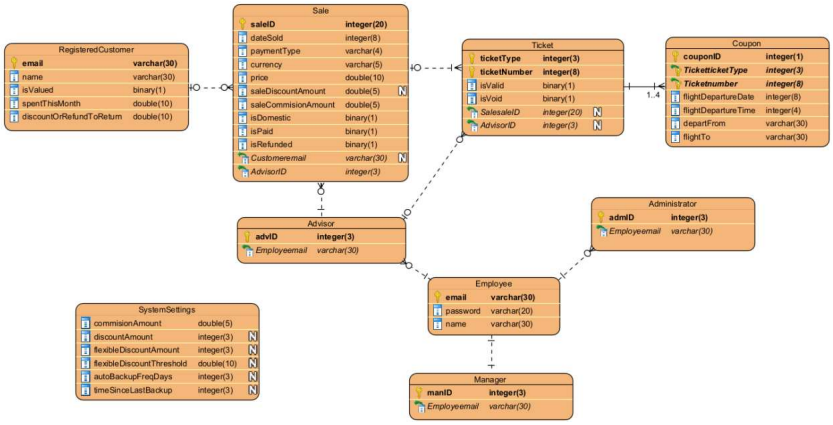
This draft was worked on by Joel.

Second Draft of Entity-Relationship Diagram 26/02/2023 :



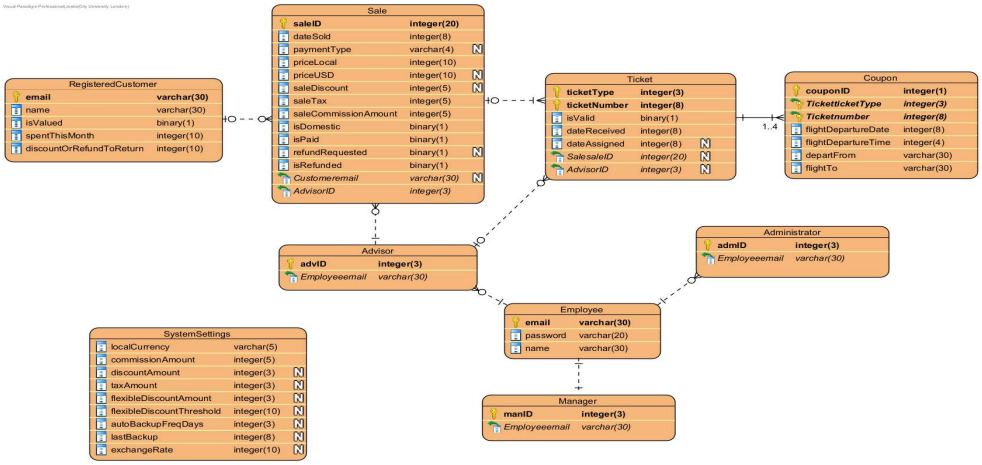
This draft was worked on by Joel.

Third Draft of Entity-Relationship Diagram 27/02/2023 :



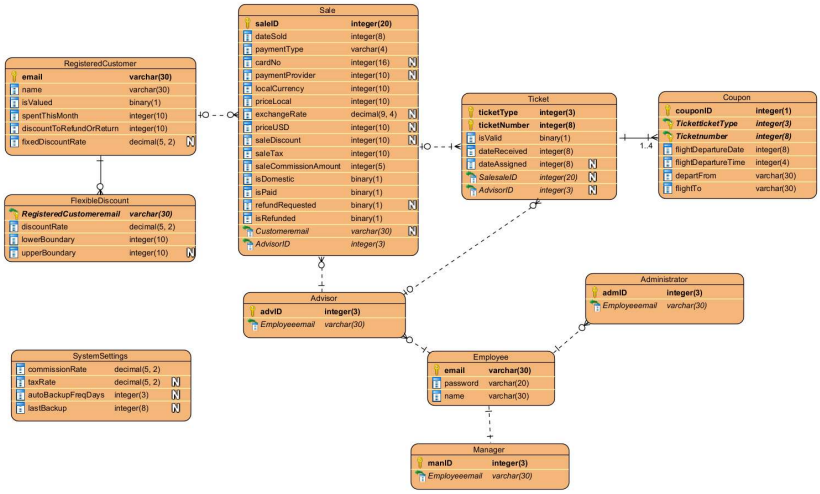
This draft was worked on by Joel.

Fourth Draft of Entity-Relationship Diagram 04/03/2023 :



This draft was worked on by Joel.

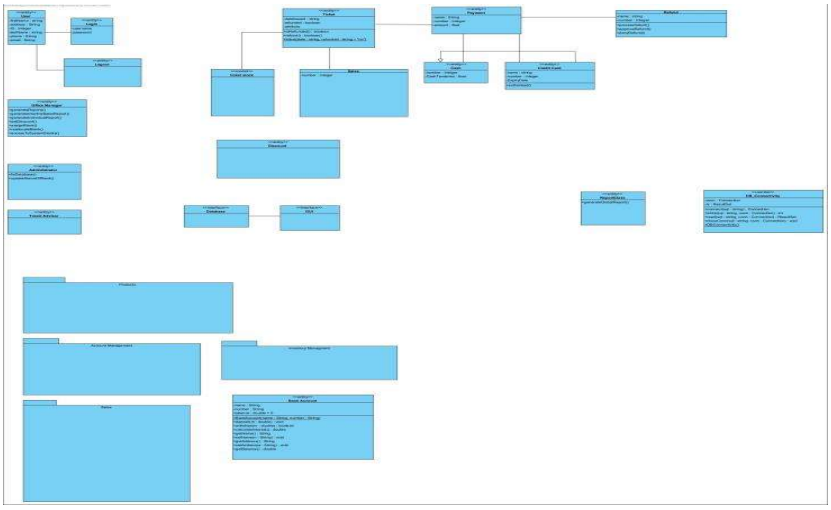
Fifth Draft of Entity-Relationship Diagram 27/03/2023 :



This draft was worked on by Joel after feedback from martins lectures

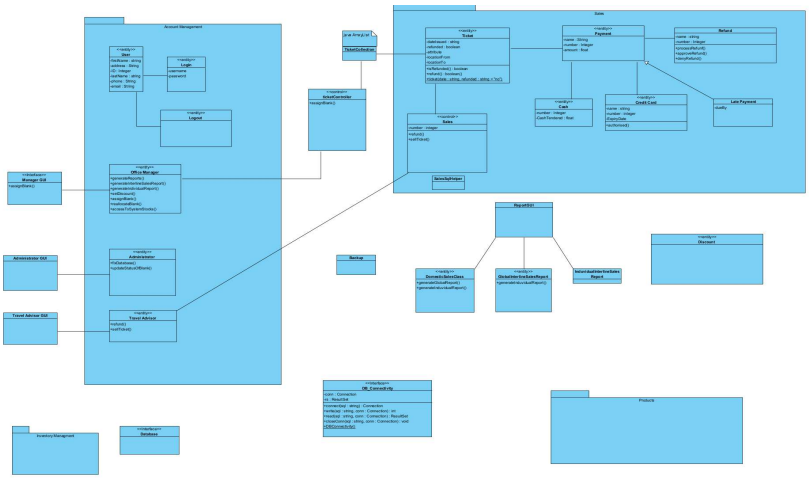
Design Class Diagrams

First Draft of Design Class Diagram 24/02/2023 :



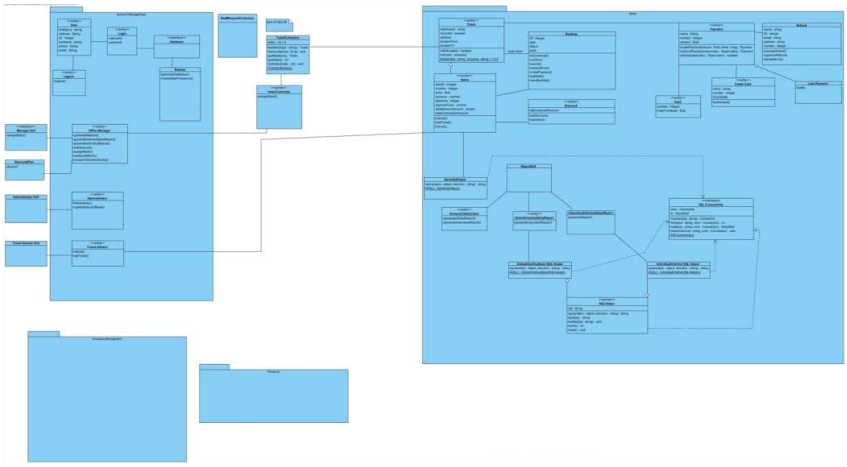
This draft was worked on by Joel and Methusha.

Second Draft of Design Class Diagram 25/02/2023 :



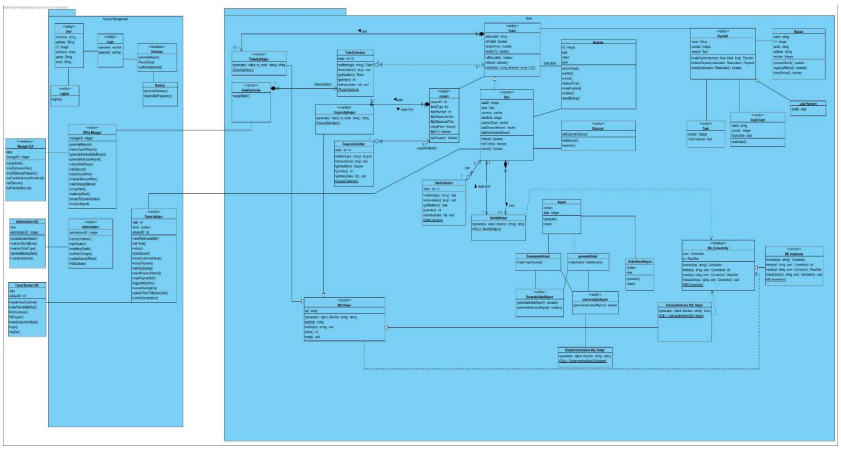
This draft was worked on by Joel and Methusha.

Third Draft of Design Class Diagram 26/02/2023 :



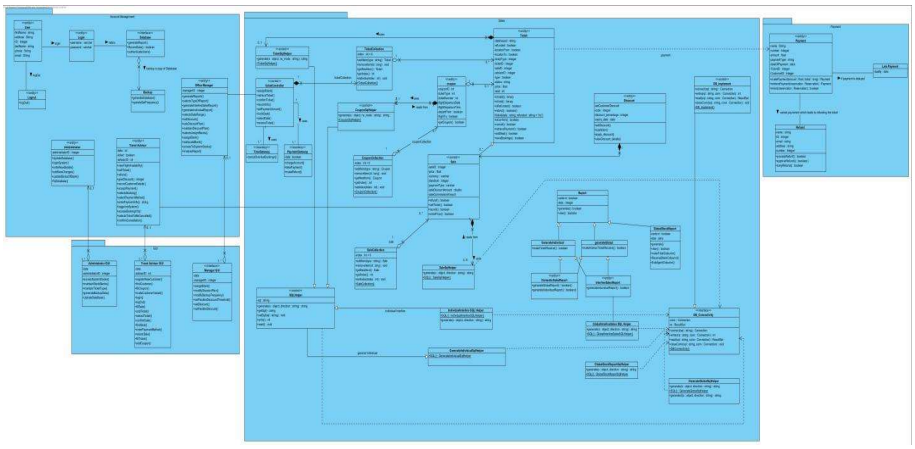
This draft was worked on by Joel and Methusha.

Fourth Draft of Design Class Diagram 27/02/2023 :



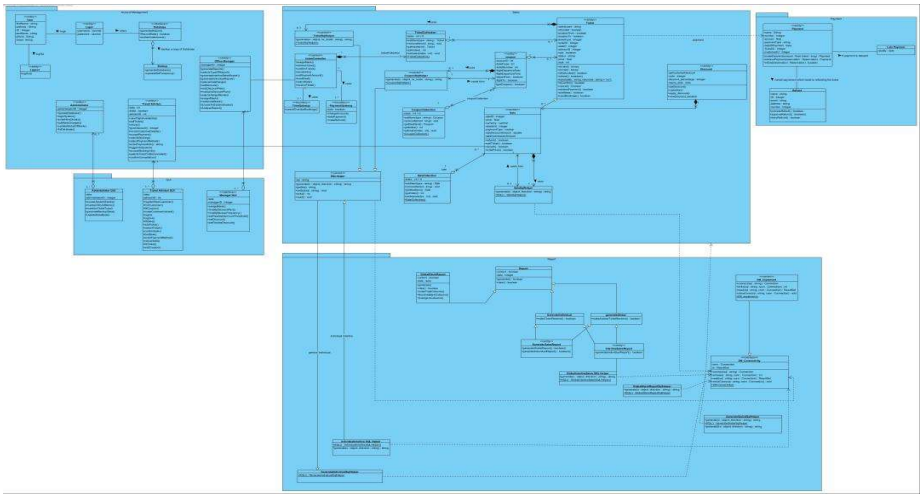
This draft was worked on by Joel and Methusha.

Fifth Draft of Design Class Diagram 28/02/2023 :



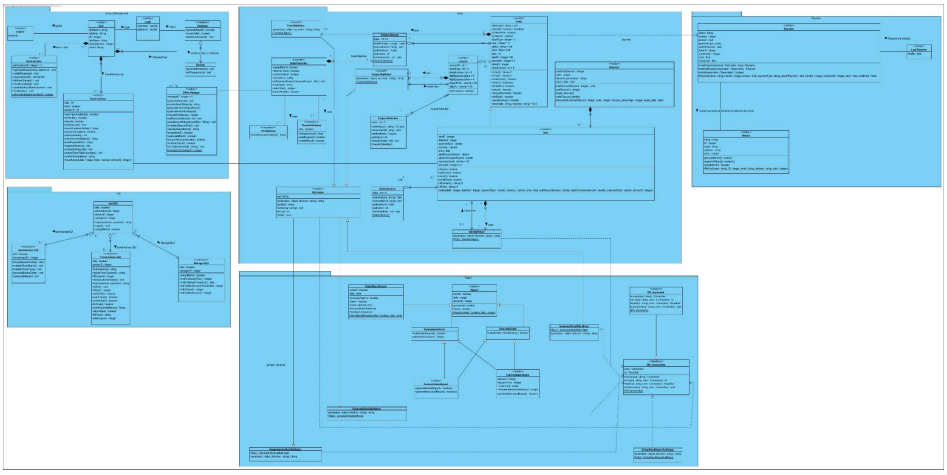
This draft was worked on by Methusha.

Sixth Draft of Design Class Diagram 01/03/2023 :



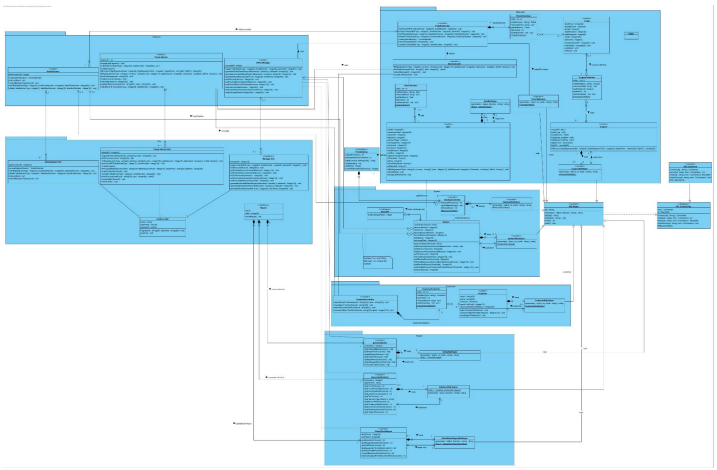
This draft was worked on by Methusha.

Seventh Draft of Design Class Diagram 03/03/2023 :



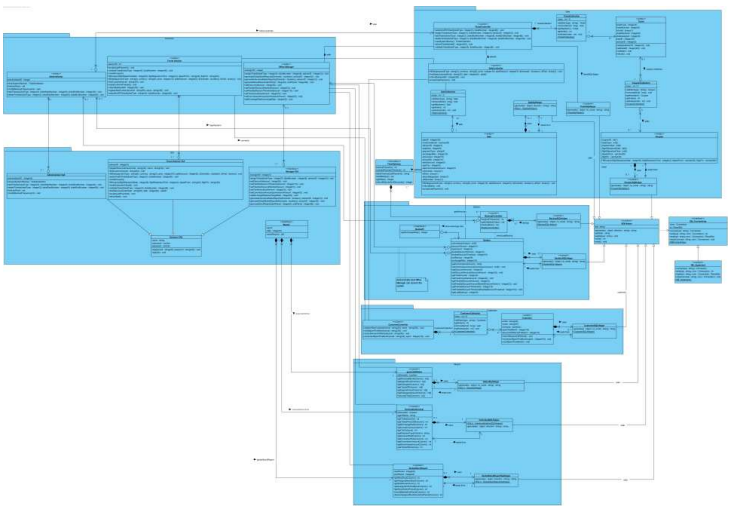
This draft was worked on by Methusha.

Eighth Draft of Design Class Diagram 04/03/2023 :



This draft was worked on by Joel and Methusha.

Final Draft of Design Class Diagram 05/03/2023 :



This draft was worked on by Joel and Methusha.

SQL Queries

First Draft of SQL Queries 01/03/2023 :

```sql
/* CREATE TABLE STATEMENTS

Pattern:
CREATE TABLE Name(
Keys,

Attributes,

Key Definitions
);

*/

CREATE TABLE Employee(
    email varchar(30),

    password varchar(20) NOT NULL,
    name varchar(30) NOT NULL,

    PRIMARY KEY (email)
);

CREATE TABLE Administrator(
    admID integer(3),
    email varchar(30) NOT NULL,

    PRIMARY KEY (admID),
    FOREIGN KEY (email) REFERENCES Employee (email)
);

CREATE TABLE Manager(
    manID integer(3),
    email varchar(30) NOT NULL,

    PRIMARY KEY (manID),
    FOREIGN KEY (email) REFERENCES Employee (email)
);

CREATE TABLE Advisor(
    advID integer(3),
    email varchar(30) NOT NULL,

    PRIMARY KEY (advID),
    FOREIGN KEY (email) REFERENCES Employee (email)
);

CREATE TABLE RegisteredCustomer(
    email varchar(30),

    name varchar(30) NOT NULL,
    isValued tinyint(1) NOT NULL DEFAULT 0,
    spentThisMonth bigint(8) NOT NULL DEFAULT 0,
    discountOrRefundToReturn bigint(8) NOT NULL DEFAULT 0,

    PRIMARY KEY (email)
);

CREATE TABLE Sale(
    saleID integer(20) AUTO_INCREMENT,
    advisorID integer(3) NOT NULL,
    customerEmail varchar(30) NULL,

    dateSold date,
    paymentType varchar(4) NULL,
    priceLocal varchar(5),
    priceUSD integer(20),
```

```sql
    saleDiscountAmount integer(4),
    taxAmount integer(4),
    saleCommissionAmount integer(4),
    isDomestic tinyint(1),
    isPaid tinyint(1) DEFAULT 0,
    refundRequested tinyint(1) DEFAULT 0,
    isRefunded tinyint(1) DEFAULT 0,

    PRIMARY KEY (saleID),
    FOREIGN KEY (advisorID) REFERENCES Advisor (advID),
    FOREIGN KEY (customerEmail) REFERENCES RegisteredCustomer (email)
);

CREATE TABLE Ticket(
    ticketType integer(3),
    ticketNumber integer(8),
    advisorID integer(3),
    saleID integer(20),

    isValid tinyint(1) DEFAULT 0,
    dateReceived date,
    dateAssigned date,

    UNIQUE (ticketType, ticketNumber),
    PRIMARY KEY (ticketType, ticketNumber),
    FOREIGN KEY (advisorID) REFERENCES Advisor (advID),
    FOREIGN KEY (saleID) REFERENCES Sale (saleID)
);

CREATE TABLE Coupon(
    CouponID integer(1) AUTO_INCREMENT,
    ticketType integer(3),
    ticketNumber integer(8),

    flightDepartureDate date,
    flightDepartureTime integer(4),
    departFrom varchar(30),
    flightTo varchar(30),

    UNIQUE (CouponID, ticketType, ticketNumber),
    PRIMARY KEY  (CouponID, ticketType, ticketNumber),
    FOREIGN KEY (ticketType, ticketNumber) REFERENCES Ticket (ticketType, ticketNumber)
);

CREATE TABLE SystemSettings(
    localCurrency varchar(5),
    commissionAmount integer(4),
    taxAmount integer(4),
    discountAmount integer(4),
    flexibleDiscountAmount integer(4),
    flexibleDiscountThreshold integer(8),
    autoBackupFreqDays integer(3) DEFAULT 7,
    lastBackup date,
    exchangeRate integer(20)
);

/*Note settings exchange rate is the exchange rate for the day,
not the same as sale exchange rate - which is for the sale and is saved for refunds*/
```

```sql

/* Other assign statements for reports */

INSERT INTO RegisteredCustomer VALUES
    ("dylan@gmail.com", "Dylan Dylandson", False, 0, 0);

INSERT INTO Employee VALUES
    ("bob@gmail.com", "password", "bob"),
    ("dave@gmail.com", "password", "bob");

INSERT INTO Advisor VALUES
    (1, "bob@gmail.com"),
    (2, "dave@gmail.com");

INSERT INTO Sale(advisorID,
    customerEmail,
    dateSold,
    paymentType,
    priceLocal,
    saleDiscountAmount, taxAmount, saleCommissionAmount,
    isDomestic, isPaid,
    refundRequested, isRefunded) VALUES
    (1, NULL, "2023-02-22", "card",
        20000,
        1000, 2000, 500,
        False, True,
        False, False),
    (1, NULL, "2023-02-22", "card",
        23000,
        1500, 2000, 500,
        False, True,
        False, False),
    (1, NULL, "1990-02-22", "card",
        16000,
        1500, 2000, 500,
        False, True,
        False, False),
    (2, NULL, "2023-02-22", "card",
        20000,
        1000, 2000, 500,
        False, False,
        False, False);

```

```sql
/* INSERT STATEMENTS */
/*new ticket stocks added to system*/
INSERT INTO Ticket (ticketType, ticketNumber, advisorId, saleID, isValid, dateReceived, dateAssigned) VALUES
    (444, 1, NULL, NULL, False, "2023-02-15", NULL),
    (444, 2, 1, 1, True, "2023-02-15", "2023-02-20"),
    (444, 3, 2, NULL, False, "2023-02-15", "2023-02-20"),
    (444, 4, NULL, NULL, False, "2023-02-15", NULL),
    (451, 1, 1, 1, True, "2023-02-15", "2023-02-20"),
    (444, 5, 2, 2, True, "2023-02-15", "2023-02-20"),
    (444, 6, 2, 3, True, "2023-02-15", "2023-02-20"),
    (444, 7, 1, 2, True, "2023-02-15", "2023-02-20"),
    (444, 8, NULL, NULL, False, "2023-02-15", NULL),
    (451, 2, 2, 3, True, "2023-02-15", "2023-02-20"),
    (444, 9, 2, NULL, True, "2023-02-15", "2023-02-20"),
    (444, 10, 1, NULL, True, "2023-02-15", "2023-02-20"),
    (444, 11, NULL, NULL, True, "2023-02-15", NULL),
    (444, 12, NULL, NULL, False, "2023-02-15", NULL),
    (451, 3, NULL, NULL, True, "2023-02-15", NULL);



/* add coupons for international ticket with 3 legs for journey */
INSERT INTO Coupon(ticketType, ticketNumber, flightDepartureDate, flightDepartureTime, departFrom, flightTo) VALUES
    (444, 00000007, "2023-05-20", 1230, 'London','Paris'),
    (444, 00000007, "2023-05-20", 1500, 'Paris','Berlin'),
    (444, 00000007, "2023-05-20", 1915, 'Berlin','Amsterdam');
```

```sql

/* UPDATE STATEMENTS */

/* Assign 3 tickets to an Advisor */
UPDATE Ticket
SET
    advisorID = 001,
    dateAssigned = NOW()
WHERE (ticketType = 444) AND (ticketNumber BETWEEN 1 AND 5);


/* Sell 3 tickets*/
UPDATE Ticket t, Sale s
SET t.isValid = True,
    t.saleID = 3,
    s.dateSold = NOW()
WHERE (ticketType = 444) AND (ticketNumber = 1 OR ticketNumber = 2);

UPDATE Ticket t, Sale s
SET t.isValid = True,
    t.saleID = 2,
    s.dateSold = NOW()
WHERE (ticketType = 444) AND ticketNumber = 3;

```

```sql
/* SELECT STATEMENTS */

/* select tickets without an advisor assigned to them */
SELECT ticketType, ticketNumber
FROM Ticket
WHERE advisorID IS NULL;

/* select overdue late payment sales*/
SELECT saleId, customerEmail, advisorID, dateSold
FROM Sale
WHERE (DATEDIFF(NOW(), dateSold) > 30);

```

```sql
/* STATEMENTS FOR Induvidual Interline SALES REPORT for advisor ID 1,
period 2023-02-01 to 2023-04-01*/
/*Induvidual sales info */
SELECT
    s.saleID, s.isPaid, s.paymentType, s.dateSold,
    s.priceLocal / 100
        AS 'Price',
    ((s.priceLocal / 100) - ((s.priceLocal / 100) * (s.saleDiscountAmount / 10000)))
        AS 'Discounted Price',
    ((s.priceLocal / 100) * (s.saleDiscountAmount / 10000))
        AS 'Discount',
    (((s.priceLocal / 100) - ((s.priceLocal / 100)
        * (s.saleDiscountAmount / 10000))) * (s.taxAmount
 / 10000))
        AS 'Tax',
    (((s.priceLocal / 100) - ((s.priceLocal / 100)
        * (s.saleDiscountAmount / 10000))) * (s.saleCommissionAmount / 10000))
        AS 'Commission',
    (((s.priceLocal / 100) - ((s.priceLocal / 100)
            * (s.saleDiscountAmount / 10000)))
        - (((s.priceLocal / 100) - ((s.priceLocal / 100)
            * (s.saleDiscountAmount / 10000)))
                * (s.saleCommissionAmount / 10000)))
        AS 'Remaining Agents Amount'
FROM Sale s, Ticket t
WHERE
    t.advisorID = s.advisorID
    AND s.advisorID = 1
    AND t.isValid = True
    AND s.isDomestic = False
    AND s.isRefunded = False
    AND ("2023-02-01" <= s.dateSold)
    AND (s.dateSold <= "2023-04-01")
    GROUP BY s.saleID;

/*Sales sums,
period 2023-02-01 to 2023-04-01*/
SELECT
    COUNT(*) AS 'Tickets Sold by Advisor'
FROM Sale s
INNER JOIN Ticket t ON s.saleID = t.saleID
WHERE
    t.advisorID = 1
    AND t.saleID IS NOT NULL
    AND t.advisorID = s.advisorID
    AND t.isValid = True
    AND s.isDomestic = False
    AND s.isRefunded = False
    AND ("2023-02-01" <= s.dateSold)
    AND (s.dateSold <= "2023-04-01")
    GROUP BY t.advisorID;

SELECT
    SUM(s.priceLocal / 100)
        AS 'Total Price',
    SUM((s.priceLocal / 100) - ((s.priceLocal / 100) * (s.saleDiscountAmount / 10000)))
        AS 'Total Discounted Price',
    SUM((s.priceLocal / 100) * (s.saleDiscountAmount / 10000))
        AS 'Total Discount',
    SUM(((s.priceLocal / 100) - ((s.priceLocal / 100)
        * (s.saleDiscountAmount / 10000))) * (s.taxAmount
 / 10000))
        AS 'Total Tax',
    SUM(((s.priceLocal / 100) - ((s.priceLocal / 100)
        * (s.saleDiscountAmount / 10000))) * (s.saleCommissionAmount / 10000))
        AS 'Total Commission',
    SUM(((s.priceLocal / 100) - ((s.priceLocal / 100)
```

```sql
                * (s.saleDiscountAmount / 10000)))
            - (((s.priceLocal / 100) - ((s.priceLocal / 100)
                * (s.saleDiscountAmount / 10000)))
                    * (s.saleCommissionAmount / 10000)))
            AS 'Total Agents Remaining Amount'
 FROM Sale s
 WHERE
     s.advisorID = 1
     AND s.isDomestic = False
     AND s.isRefunded = False
     AND ("2023-02-01" <= s.dateSold)
     AND (s.dateSold <= "2023-04-01")
 ;
```

```sql

/* Column 1a STATEMENTS FOR TICKET TUROVER REPORT */
/* All tickets received in
period 2023-02-01 to 2023-04-01*/
SELECT
    ticketType, ticketNumber, dateReceived
FROM Ticket
WHERE
    ("2023-02-01" <= dateReceived)
    AND (dateReceived <= "2023-04-01")
ORDER BY ticketType;

/*Total ^^ */
SELECT
    COUNT(*)
FROM Ticket
WHERE
    ("2023-02-01" <= dateReceived)
    AND (dateReceived <= "2023-04-01");



/* Column 1b All tickets received and assigned in
period 2023-02-01 to 2023-04-01*/
SELECT
    advisorID, ticketType, ticketNumber, dateReceived, dateAssigned
FROM Ticket
WHERE
    ("2023-02-01" <= dateReceived)
    AND (dateReceived <= "2023-04-01")
    AND ("2023-02-01" <= dateAssigned)
    AND (dateAssigned <= "2023-04-01")
ORDER BY ticketType;

/*Total ^^ */
SELECT
    COUNT(*)
FROM Ticket
WHERE
    ("2023-02-01" <= dateReceived)
    AND (dateReceived <= "2023-04-01")
    AND ("2023-02-01" <= dateAssigned)
    AND (dateAssigned <= "2023-04-01")
ORDER BY advisorID, ticketType;

```

```sql

/* Column 2a All tickets assigned to advisors within
period 2023-02-01 to 2023-04-01*/
SELECT
    advisorID, ticketType, ticketNumber, dateAssigned
FROM Ticket
WHERE
    ("2023-02-01" <= dateAssigned)
    AND (dateAssigned <= "2023-04-01")
ORDER BY ticketType;


/*Total ^^ */
SELECT
    COUNT(*)
FROM Ticket
WHERE
    ("2023-02-01" <= dateAssigned)
    AND (dateAssigned <= "2023-04-01")
ORDER BY advisorID, ticketType;


/* Column 2b All tickets used within
period 2023-02-01 to 2023-04-01*/
SELECT
    t.ticketType, t.ticketNumber, s.dateSold
FROM Ticket t, Sale s
WHERE
    t.advisorID = s.saleID
    AND s.isRefunded = False
    AND t.isValid = True
    AND("2023-02-01" <= s.dateSold)
    AND (s.dateSold <= "2023-04-01")
ORDER BY t.ticketType, t.ticketNumber;

/*Total ^^ */
SELECT
    COUNT(*)
FROM Ticket t, Sale s
WHERE
    t.advisorID = s.saleID
    AND s.isRefunded = False
    AND t.isValid = True
    AND("2023-02-01" <= s.dateSold)
    AND (s.dateSold <= "2023-04-01");



/* Column 3a All tickets that are unsold and available
period 2023-02-01 to 2023-04-01*/
SELECT
    ticketType, ticketNumber
FROM Ticket t, Sale s
WHERE
    (t.saleID IS NULL
    OR (s.dateSold <= "2023-04-01"))
GROUP by t.ticketType, t.ticketNumber
ORDER BY t.ticketType;

/*Total ^^ */
SELECT DISTINCT
    COUNT(DISTINCT(ticketNumber), ticketType)
FROM Ticket t, Sale s
WHERE
    (t.saleID IS NULL
    OR (s.dateSold <= "2023-04-01"));

```

```sql
/* Column 3b All tickets that are unsold and assigned by type
period 2023-02-01 to 2023-04-01*/
SELECT
    t.advisorID, ticketType, ticketNumber
FROM Ticket t, Sale s
WHERE
    (t.saleID IS NULL
    OR (s.dateSold <= "2023-04-01"))
    AND t.advisorID IS NOT NULL
GROUP by t.advisorID, t.ticketType, t.ticketNumber
ORDER BY t.advisorID, t.ticketType;

/* Detailed unsold assigned tickets */
SELECT
    COUNT(DISTINCT(ticketNumber), ticketType)
FROM Ticket t, Sale s
WHERE
    (t.saleID IS NULL
    OR (s.dateSold <= "2023-04-01"))
    AND t.advisorID IS NOT NULL;



/* DELETE STATEMENTS */

/* a customer would like their account to be deleted */
DELETE FROM RegisteredCustomer
WHERE email = 'dylan@gmail.com';

/* delete coupons made in error for ticket 444 00000003 */
DELETE FROM Coupon
WHERE ticketType = 444 AND ticketNumber = 00000007;



/* DROP TABLES */
/*
DROP TABLE Coupon;
DROP TABLE Ticket;
DROP TABLE Sale;
DROP TABLE RegisteredCustomer;
DROP TABLE Advisor;
DROP TABLE Manager;
DROP TABLE Administrator;
DROP TABLE Employee;
DROP TABLE SystemSettings;
*/
```

This draft was worked on by Joel.

Final Draft of SQL Queries 05/03/2023 :

```sql
/* CREATE TABLE STATEMENTS

Pattern:
CREATE TABLE Name(
KEYS,

Attributes,

Key Definitions
);

*/

CREATE TABLE Employee(
    email varchar(30),

    password varchar(20) NOT NULL,
    name varchar(30) NOT NULL,

    PRIMARY KEY (email)
);

CREATE TABLE Administrator(
    admID integer(2),
    email varchar(30) NOT NULL,

    PRIMARY KEY (admID),
    FOREIGN KEY (email) REFERENCES Employee (email)
);

CREATE TABLE Manager(
    manID integer(3),
    email varchar(30) NOT NULL,

    PRIMARY KEY (manID),
    FOREIGN KEY (email) REFERENCES Employee (email)
);

CREATE TABLE Advisor(
    advID integer(3),
    email varchar(30) NOT NULL,

    PRIMARY KEY (advID),
    FOREIGN KEY (email) REFERENCES Employee (email)
);

CREATE TABLE RegisteredCustomer(
    email varchar(30),

    name varchar(30) NOT NULL,
    isvalued tinyint(1) NOT NULL DEFAULT 0,
    spentThisMonth bigint(6) NOT NULL DEFAULT 0,
    discountOrRefundToReturn bigint(9) NOT NULL DEFAULT 0,

    PRIMARY KEY (email)
);

CREATE TABLE Sale(
    saleID integer(20) AUTO_INCREMENT,
    advisorID integer(3) NOT NULL,
    customerEmail varchar(30) NULL,

    dateSold date,
    paymentType varchar(4),
    currency varchar(5),
    price integer(20),
    saleDiscountAmount integer(4),
    saleTaxAmount integer(4),
    saleCommissionAmount integer(4),
    isDomestic tinyint(1),
    isPaid tinyint(1) DEFAULT 0,
    isRefunded tinyint(1) DEFAULT 0,

    PRIMARY KEY (saleID),
    FOREIGN KEY (advisorID) REFERENCES Advisor (advID),
    FOREIGN KEY (customerEmail) REFERENCES RegisteredCustomer (email)
);

CREATE TABLE Tickets(
    ticketType integer(1),
    ticketNumber integer(8),
    advisorID integer(3),
    saleID integer(20),

    isvalid tinyint(1) DEFAULT 0,
    seats integer(1) DEFAULT 1,

    UNIQUE (ticketType, ticketNumber),
    PRIMARY KEY (ticketType, ticketNumber),
    FOREIGN KEY (advisorID) REFERENCES Advisor (advID),
    FOREIGN KEY (saleID) REFERENCES Sale (saleID)
);
```

```
CREATE TABLE Coupon(
    CouponID integer(11) AUTO_INCREMENT,
    ticketType integer(3),
    ticketNumber integer(8),

    flightDepartureDate date,
    flightDepartureTime integer(4),
    departfrom varchar(30),
    flightto varchar(30),

    UNIQUE (CouponID, ticketType, ticketNumber),
    PRIMARY KEY  (CouponID, ticketType, ticketNumber),
    FOREIGN KEY (ticketType, ticketNumber) REFERENCES Ticket (ticketType, ticketNumber)
);

CREATE TABLE SystemSettings(
    commissionAmount integer(4),
    saleTaxamount integer(4),
    discountAmount integer(4),
    flexibleDiscountAmount integer(4),
    flexibleDiscountThreshold integer(4),
    autobackupfreqDays integer(3) DEFAULT 7,
    lastBackup date
);



/* INSERT STATEMENTS */

/*New ticket stocks added to system*/
INSERT INTO Ticket VALUES
    (444, 00000002, NULL, NULL, False, 1),
    (444, 00000002, NULL, NULL, False, 2),
    (444, 00000002, NULL, NULL, False, 3),
    (444, 00000004, NULL, NULL, False, 4),
    (451, 00000001, NULL, NULL, False, 21);

/*5 add coupons for international ticket with 5 legs for Survey */
INSERT INTO Coupon(ticketType, ticketNumber, flightDepartureDate, flightDepartureTime, departfrom, flightto) VALUES
    (444, 00000001, 20012023, 1230, 'London', 'Paris'),
    (444, 00000001, 20012023, 1100, 'Paris','Berlin'),
    (444, 00000001, 20012023, 1915, 'Berlin','Amsterdam');

/* Other assign statements for reports */
INSERT INTO Employee VALUES
    ('bob@gmail.com', 'password', 'bob');

INSERT INTO Adviser VALUES
    (1, 'bob@gmail.com');

INSERT INTO Sale(advisorID, customerEmail, dateSold, paymentType, currency, price, saleDiscountAmount, saleTaxAmount, saleCommissionAmount, isDomestic, isPaid, isRefunded) VALUES
    (1, NULL, '2022-11-02', 'card', 'USD', 20000, 3000, 2000, 500, False, True, False),
    (1, NULL, '1990-01-01', 'cash', 'USD', 20000, 1000, 2000, 500, False, True, False);



/* UPDATE STATEMENTS */

/* Assign 3 tickets to an Adviser */
UPDATE Ticket
SET
    advisorID = 001
WHERE (ticketType = 444) AND (ticketNumber BETWEEN 1 AND 4);

/* Sell 3 tickets*/
UPDATE Ticket
SET isValid = True,
    saleID = 1
WHERE (ticketType = 444) AND (ticketNumber = 1 OR ticketNumber = 2);

UPDATE Ticket
SET isValid = True,
    saleID = 2
WHERE (ticketType = 444) AND ticketNumber = 3;



/* SELECT STATEMENTS */

/* select tickets without an advisor assigned to them */
SELECT ticketType, ticketNumber
FROM Ticket
WHERE advisorID IS NULL;

/* select overdue late payment sales*/
SELECT saleId, customeremail, advisorID, datesold
FROM Sale
WHERE (DATEDIFF(NOW(), datesold) > 30);
```

```
/* DELETE STATEMENTS */

/* a customer would like their account to be deleted */
DELETE FROM RegisteredCustomer
WHERE email = 'steve@gmail.com';

/* delete coupons made in error for ticket 444 00000003 */
DELETE FROM Coupon
WHERE ticketType = 444 AND ticketNumber = 00000003;



/* STATEMENTS FOR Individual Interline SALES REPORT for advisor ID 1 */
/*Individual sales info */
SELECT
    s.saleID, s.isPaid, s.paymentType,
    s.price / 100
        AS 'Price',
    ((s.price / 100) - ((s.price / 100) * (s.saleDiscountAmount / 10000)))
        AS 'Discounted Price',
    ((s.price / 100) * (s.saleDiscountAmount / 10000))
        AS 'Discount',
    (((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleCommissionAmount / 10000))
        AS 'Commission Amount',
    (((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleTaxAmount / 10000))
        AS 'Taxamount',
    (s.price / 100
        - (((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleCommissionAmount / 10000))
        - (((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleTaxAmount / 10000)))
        AS 'Remaining Monaccess Amount'
FROM Sale s, Ticket t
WHERE
    t.advisorID = s.advisorID
    AND t.advisorID = 1
    AND t.isValid = 1
    AND s.isDomestic = False
    AND s.isRefunded = False
    GROUP BY s.saleID;

/*Sales sums */
SELECT
    COUNT(*) AS 'Tickets Sold by Advisor'
FROM Ticket t, Sale s
WHERE

    t.advisorID = 1
    AND t.saleID IS NOT NULL
    AND t.advisorID = s.advisorID
    AND t.isValid = 1
    AND s.isDomestic = False
    AND s.isRefunded = False
    GROUP BY t.advisorID;

SELECT
    sum(s.price / 100)
        AS 'Total Price',
    sum((s.price / 100) - ((s.price / 100) * (s.saleDiscountAmount / 10000)))
        AS 'Total Discounted Price',
    sum((s.price / 100) * (s.saleDiscountAmount / 10000))
        AS 'Total Discount',
    sum(((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleCommissionAmount / 10000))
        AS 'Total Commission Amount',
    sum(((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleTaxAmount / 10000))
        AS 'Total Taxamount',
    sum(s.price / 100
        - (((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleCommissionAmount / 10000))
        - (((s.price / 100) - ((s.price / 100)
            - (s.saleDiscountAmount / 10000))) * (s.saleTaxAmount / 10000)))
        AS 'Total Remaining Monaccess Amount'
FROM Sale s
JOIN Ticket t ON s.saleID = t.saleID
WHERE
    t.advisorID = 1
    AND s.saleID = t.saleID
    AND t.advisorID = s.saleID
    AND t.isValid = 1
    AND s.isDomestic = False
    AND s.isRefunded = False
GROUP BY t.advisorID;
```

This draft was worked on by Joel.

Use Case Specifications

Use Case 1: Sell Ticket

ID: TM01	Use Case: Sell Ticket
Brief description: The customer makes a phone inquiry for a domestic flight ticket and wants to do the payment by card.	
Primary actor: Travel Advisor	
Secondary actor: None	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in and viewing domestic flight availability.	
Flow of events: 1. Travel advisor searches through the list of available domestic flights and selects a flight for the customer. 2. Travel advisor confirms availability of seats for the selected flight. 3. Travel advisor checks if the customer already has a registered account and if they are a valued customer. 4. If the customer is already registered in the database 4.1. The advisor proceeds to the next step : payment Else If the customer account doesn't exist in the database 4.2. The advisor creates a new customer account (enters customer information, including name, address, and payment details) and proceeds to payment 5. Payment is received and is recorded in the system by the travel advisor 6. Ticket is assigned to the customer. 7. System updates the Sale with the new customer information and reservation details.	
Postconditions: 1. The ticket is sold to the customer and the flight is updated with the new reservation information. 2. The database is updated with the relevant information.	

Alternative flow: PaymentInformationInvalid CustomerInformationIncomplete PaymentFailure

Alternative flow: PaymentInformationInvalid
ID: TM01.1
Brief description: If the customer's payment information is invalid or insufficient, the travel agent informs the customer and requests a different payment method.
Primary actor: Travel Advisor
Secondary actor: None
Preconditions: 1. The provided payment information is invalid or incorrect.
Alternative flow: 4. The system notifies the advisor that the information provided is incorrect. 5. The advisor requests correct details from the customer. 6. The advisor updates the details and proceeds with the next steps.
Postconditions: None

Use Case 2: Assign Blanks

ID: TM02	Use Case: Assign Blanks
Brief description: The Office Manager assigns a set of blanks to a Travel Agent for booking purposes.	
Primary actor: Office Manager	
Secondary actor: None	
Preconditions: 1. The Office Manager is logged in and has access to the system 2. and there are available blanks to assign.	

Flow of events: 1. The Office Manager selects the "Assign Blanks" option from the system menu. 2. The system presents a list of available blanks to assign. 3. The Office Manager selects the desired set of blanks to assign. 4. The system presents a list of registered Travel Advisors. 5. The Office Manager selects the Travel Advisor to assign the blanks to. 6. The system registers the assigned blanks to the selected Travel advisor and updates the availability of the blanks in the system.
Postconditions: 1. The selected blanks are assigned to the selected Travel advisor and are no longer available for other Travel advisors to book. 2. The system updates the availability of the assigned blanks in the system.
Alternative flow: NoRegisteredTravelAdvisor NoAvailableBlanksToAssign

Alternative flow: NoRegisteredTravelAdvisor
ID: TM02.1
Brief description: There are no registered Travel advisors.
Primary actor: Office Manager
Secondary actor: None
Preconditions: The system does not contain a valid TravelAdvisorAccount.
Alternative flow: 4. The system prompts the actor that no valid TravelAdvisorAccount exists. 5. The actor acknowledges the notification. 6. The actor registers a new TravelAdvisor Account and proceeds with the operation.
Postconditions: 1. There is a new Travel Advisor account registered. 2. The database is updated.

Alternative flow: NoAvailableBlanksToAssign
ID: TM02.2
Brief description: There are no available blanks to assign.
Primary actor: Office Manager
Secondary actor: None
Preconditions: The system does not contain blanks to assign.
Alternative flow: 2. The system prompts the actor that there are no valid blanks to assign. 5. The actor acknowledges the notification. 6. The system removes the session.
Postconditions: None

Use Case 3: Late Ticket Payment

ID: TM03	Use Case: Late Ticket Payment
Brief description: This use case allows a Travel Advisor to handle the situation where a customer has not paid for their ticket in time.	
Primary actor: Travel Advisor	
Secondary actor: Time	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in and has access to the customer's ticket information. 2. The customer has chosen to make a late payment while booking the ticket previously.	
Flow of events: 1. The Travel Advisor receives notification that a customer has not paid for their ticket in time (30 days). 2. The Travel Advisor contacts the customer to remind them to pay for their ticket.	

3. If the customer pays 3.1. The Travel Advisor updates the payment information in the system and confirms the ticket. Else If the customer does not pay within a given timeframe 3.2. The Travel Advisor cancels the ticket and updates the system accordingly.
Postconditions: 1. The Travel Advisor has either received payment or has cancelled the ticket. 2. The database is updated accordingly.
Alternative flow: None

Use Case 4: Update Database

ID: TM04	Use Case: Update Database
Brief description: The administrator updates the database with new information or changes existing data.	
Primary actor: Administrator	
Secondary actor: Airvia Database	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in and has appropriate permissions to update the database.	
Flow of events: 1. Administrator logs into the Airvia system and navigates to the database update page. 1. System displays a list of available data fields that can be updated. 2. Administrator selects the data field they want to update. 3. System prompts the administrator to enter the new information. 4. Administrator enters the new information and saves the changes. 5. System updates the database with the new information. 6. System displays a confirmation message to the administrator that the update was successful.	

Postconditions: 1. The database is updated with the new information or changes.
Alternative flow: InvalidDataEntered CancelledChange

Alternative flow: InvalidDataEntered
ID: TM04.1
Brief description: The Administrator enters invalid data.
Primary actor: Administrator
Secondary actor: AirVia Database
Preconditions: The system does not contain blanks to assign.
Alternative flow: 4. The system displays an error message that the entered information is not valid. 5. The system prompts the administrator to correct the input before proceeding.
Postconditions: None

Alternative flow: CancelledChange
ID: TM04.2
Brief description: The Administrator cancelled the update.
Primary actor: Administrator
Secondary actor: AirVia Database
Preconditions: The system does not contain blanks to assign.

Alternative flow: 4. The administrator cancels the update. 5. The system discards any changes made. 6. The system returns to the database update page.
Postconditions: No changes made.

Use Case 5: Record Customer Details

ID: TM05	Use Case: Record Customer Details
Brief description: The travel advisor records the personal and contact information of a new customer.	
Primary actor: Travel Advisor	
Secondary actor: None	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in and has identified a new customer.	
Flow of events: 1. Travel advisor requests the new customer's personal and contact information 2. Travel advisor enters the customer's name and any additional information into the system. 3. System validates and saves the customer details in the database.	
Postconditions: 1. Customer details are recorded in the system and can be accessed for future bookings and communications.	
Alternative flow: None	

Use Case 6: Generate Report

ID: TM06	Use Case: Generate Report
Brief description: The Office Manager generates reports for analysing sales and other data.	
Primary actor: Office Manager	
Secondary actor: None	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in and has access to the system.	
Flow of events: 1. The primary actor navigates to the database update page. 2. The Office Manager selects the type of report to generate (stock reports and individual or global variations of domestic sales report and interline sales report). 3. The Office Manager selects the date range for the report. 4. The system generates the report based on the selected criteria. 5. The report is displayed for the Office Manager to analyse.	
Postconditions: 1. Reports are generated and can be analysed.	
Alternative flow: UserNotAuthorised	

Alternative flow: UserNotAuthorised
ID: TM06.1
Brief description: Office Manager is not authorised to access the system.
Primary actor: Administrator
Secondary actor: AirVia Database
Preconditions: The manager is not authorised to access the system.

Alternative flow: 1. The manager is not authorised to access the system. 2. The system displays an error message. 3. The manager is redirected to the login page
Postconditions: None

Use Case 7: Set/Maintain Discount Plan

ID: TM07	Use Case: Set/Maintain Discount Plan
Brief description: The Office Manager sets and maintains the discount plan for customers.	
Primary actor: Office Manager	
Secondary actor: None	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in and has permission to set and maintain the discount plan.	
Flow of events: 1. The Office Manager selects the "Set/Maintain Discount Plan" option from the Airvia system menu. 2. The system displays the current discount plan. 3. The Office Manager updates the discount plan by adding, modifying or removing discounts. 4. The Airvia system validates the updated discount plan and saves it to the Airvia database.	
Postconditions: 1. The discount plan is updated and customers are able to receive applicable discounts.	
Alternative flow: UpdateError	

Alternative flow: UpdateError
ID: TM07.1
Brief description: The system encounters an error while validating the updated discount plan.
Primary actor: Administrator
Secondary actor: AirVia Database
Preconditions: The system displays an error.
Alternative flow: 4. The system displays an error message. 5. The system prompts the Office manager to correct the discount plan
Postconditions: None

Use Case 8: Cancel Ticket

ID: TM08	Use Case: Cancel Ticket
Brief description: The customer has requested a ticket cancellation.	
Primary actor: Travel Advisor	
Secondary actor: None	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in. 2. The ticket must exist in the system. 3. The ticket must not have already been cancelled.	
Flow of events: 1. The travel advisor selects the ticket to be cancelled from a list of tickets. 2. The system displays the ticket details and prompts the travel advisor to confirm the cancellation. 3. The travel advisor confirms the cancellation. 4. The system updates the ticket status to "cancelled."	

Postconditions: The ticket status is updated to "cancelled" in the system.
Alternative flow: TicketNotFound TicketAlreadyCancelled

Use Case 9: Take Payment

ID: TM09	Use Case: Take Payment
Brief description: The travel advisor accepts payment from the customer for a booking.	
Primary actor: Travel Advisor	
Secondary actor: None	
Preconditions: 1. AirVia Software is operational, the primary actor has logged in. 2. The customer has made a ticket inquiry and is ready to pay.	
Flow of events: 1. The travel advisor selects the Sale for which the payment is being made. 2. The travel advisor selects the payment method (credit/debit card, cash, etc.). 3. The travel advisor marks the Sale as paid in the system	
Postconditions: 1. Payment has been accepted and the Sale is updated.	
Alternative flow: None	

Use Case 10: Request Refund

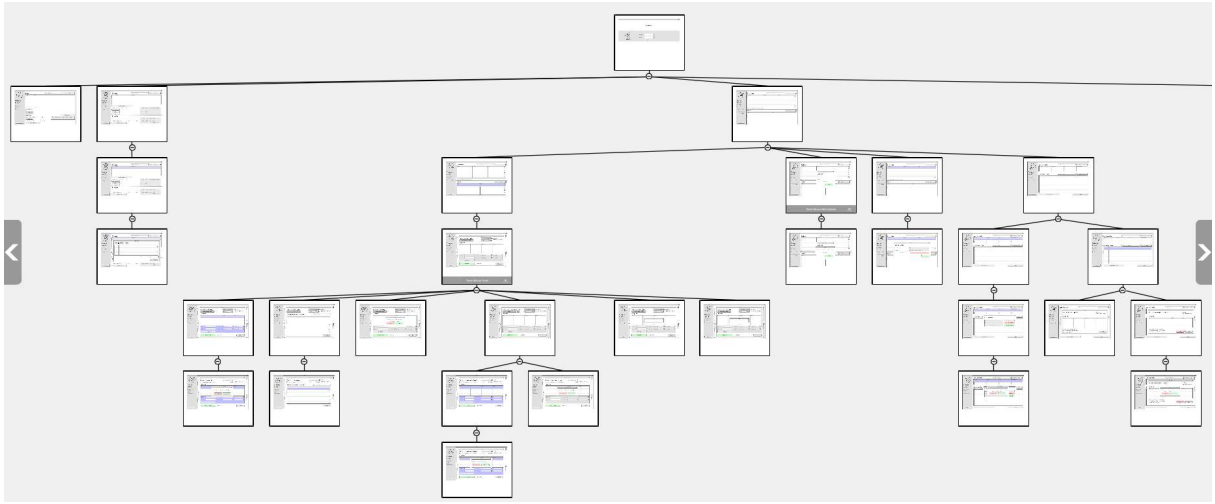
ID: TM10	Use Case: Request Refund
Brief description: The travel advisor goes through the refund process on a cancelled ticket for the customer	
Primary actor: Travel Advisor	

Secondary actor: None
Preconditions: 1. AirVia Software is operational, the primary actor has logged in. 2. The customer has made a payment for a flight ticket. 3. The booking has been cancelled or the customer is eligible for a refund. 4. The travel advisor has access to the booking and payment information.
Flow of events: 1. The travel advisor accesses the booking and payment information. 2. The system verifies the eligibility of the booking for a refund. 3. The travel advisor initiates the refund process. 4. The system calculates the refund amount to be returned based on the Sale. 5. The travel advisor returns the payment and updates the Sale to refunded.
Postconditions: 1. The customer has received the refund amount. 2. The Sale is updated to show the refund. 3. The Travel advisor has received their commission for the original ticket sale.
Alternative flow: BookingNotEligibleForRefund

This Final Draft was worked on by The Entire Team on the 4th and 5th March 2023.

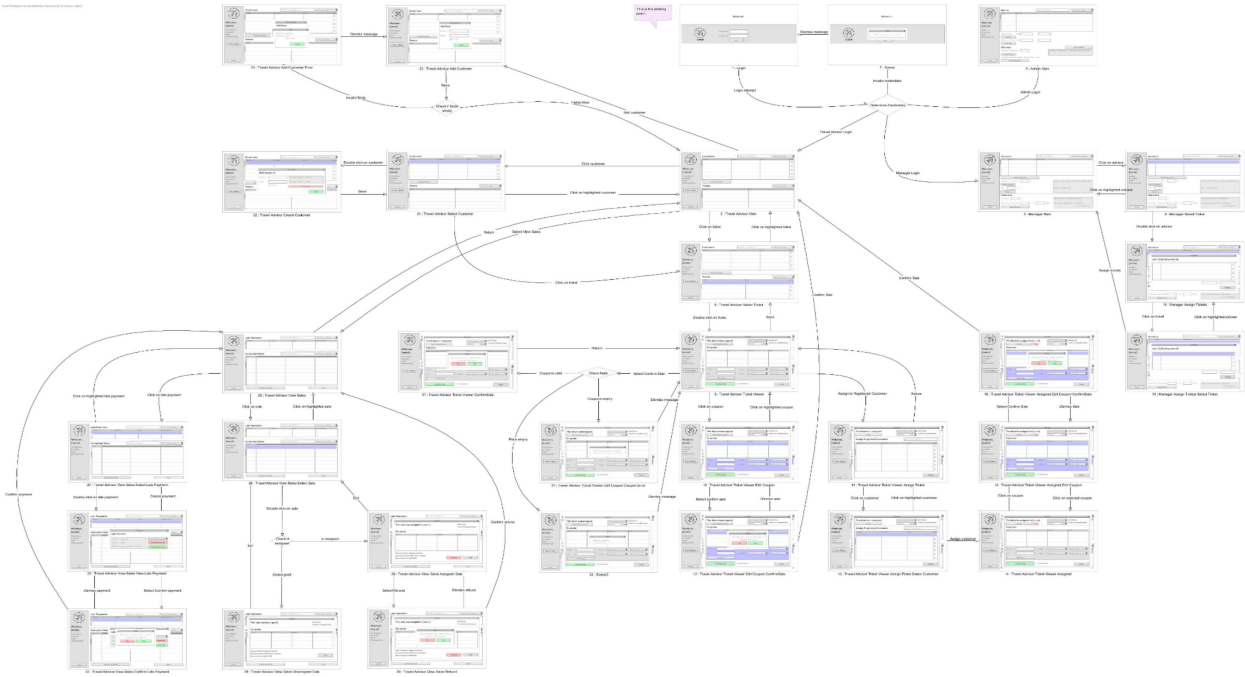
GUI Design

First Draft of GUI Design 04/03/2023 :



This draft was worked on by Tony.

Final Draft of GUI Design 05/03/2023 :



This draft was worked on by Tony.

Risk Register and Version Control

Here Attached Below is a Word Doc that displays the risk register and version control of this project binder:

Version Control And Risk Register.docx

This Document Was Done By Khalid.

Meeting 7: 6th March Time: 12am to 1pm

Agendas:

- **Discussed the first submission**
- **Discussed the Project Report**
- **Team planned to take a break**

Key points in the meeting- Discussed the first submission
We discussed the first submission and summarised each member's work. We ensured everyone was aligned with the project goals and requirements.

Key points in the meeting- Discussed the Project Report
We reviewed the project report and clarified unfamiliar sections with the respective team members. It was a productive discussion, and we were confident in our understanding of the work.

Key points in the meeting- Team planned to take a break
We decided to take a break after a successful first submission and will resume work next week. We stressed the importance of self-care and to have a little bit more social time and planned to reflect on the project during the break.

Meeting 8: 13th March Time: 12am to 1pm

Agendas:

- **Started planning the 2nd submission as a team**
- **Assign tasks to team members**

Key points in the meeting- Started planning the 2nd submission as a team
As a team, we started planning the second submission. We reviewed the project's progress and discussed the next steps needed to achieve our goals. We outlined the tasks that needed to be completed and assigned responsibilities to team members. We agreed on a timeline for completing the tasks, and we scheduled regular check-ins to track progress. The team was excited about the project's next phase and motivated to continue working collaboratively towards the common goal.

Key points in the meeting- Assign tasks to team members
During the team meeting, we assigned tasks to team members based on their expertise and interests. The programmer was assigned the majority of the Java work, given their proficiency in the language. The Tester began working on the GUI interface, leveraging their experience in software testing and design. We made sure that each team member was clear on their assigned tasks and had the necessary resources to complete their work. We emphasised the importance of collaboration and encouraged team members to seek help if they faced any difficulties. By assigning tasks based on individual strengths, we were confident that the project would progress smoothly and efficiently.

Meeting 9: 27th March Time: 12am to 1pm

Agendas:

- Project Structure of the helper and controller classes get implemented
- the programmer assigns tasks to help with the coding workload
- what output is required from the delegated tasks programmer expects

Key points in the meeting- Project Structure of the helper and controller classes get implemented

During the development phase of our project, the helper and controller classes were implemented. We recognized that a well-structured project was crucial for efficient development, maintenance, and scalability. The helper classes were designed to provide reusable methods and functions to the rest of the project. The controller classes acted as a mediator between the model and view components of the software. By implementing these classes, we were able to streamline the development process and ensure that the project remained organised and easy to maintain.

Key points in the meeting- the programmer assigns tasks to help with the coding workload

As the coding workload increased, the programmer assigned and delegated tasks to team members. By doing so, we were able to ensure that the project remained on schedule, and everyone had a manageable workload. The programmer provided clear instructions on the tasks to be completed and communicated regularly to ensure that everything was on track. The delegation of tasks also allowed the team members to leverage their individual strengths and experience, resulting in higher quality work.

Key points in the meeting- what output is required from the delegated tasks programmer expects

To save time coding the backend Java for the software, the programmer assigned SQL tasks to the delegated team members. This helped to alleviate the workload and allowed the programmer to focus on more complex coding tasks. The programmer provided clear instructions on the required output, and the team members were able to complete the SQL tasks efficiently. By delegating these tasks, the team was able to work collaboratively towards the project's goals, ensuring that everything was on track and completed to a high standard.