RAMAIAH UNIVERSITY OF APPLIED SCIENCES

Assignment

Course Code	CSC210A
Course Name	Software Development Fundamentals
Programme	B.Tech
Department	CSE
Faculty	FET

Name of the Student	Satyajit Ghana
Reg. No.	17ETCS002159
Semester/Year	04/2017
Course Leader(s)	Ms. Sahana P Shankar

Declaration Sheet

Student Name	Satyajit Ghana		
Reg. No	17ETCS002159		
Programme	B.Tech	Semester/Year	04/2017
Course Code	CSC210A		
Course Title	Software Development Fundamentals		
Course Date		to	
Course Leader	Ms. Sahana P Shankar		

Declaration

The assignment submitted herewith is a result of my own investigations and that I have conformed to the guidelines against plagiarism as laid out in the Student Handbook. All sections of the text and results, which have been obtained from other sources, are fully referenced. I understand that cheating and plagiarism constitute a breach of University regulations and will be dealt with accordingly.

Signature of the Student		Date	
Submission date stamp (by Examination & Assessment Section)			

Signature of the Course Leader and date	Signature of the Reviewer and date

Contents

List Of Figures

1 Question 1

Solution to Question No. 1 Part A

"Software Test Engineer's job is at stake with increase in emphasis on automation testing"

1.1 Introduction to the topic

Testing is an integral part of any successful software project. The type of testing (manual or automated) depends on various factors, including project requirements, budget, timeline, expertise, and suitability. Three vital factors of any project are of course time, cost, and quality – the goal of any successful project is to reduce the cost and time required to complete it successfully while maintaining quality output. When it comes to testing, one type may accomplish this goal better than the other.

(Whitney Donaldson, 2017)

With the increase of automation everywhere, the need for manual testing is decreasing. For a long time, manual testing worked very well. But today there is so much preponderance of software that it is just not feasible for humans to test all the software manually. This is where the software testing automation takes over. There are a lot of advantages of automated testing process thanks to its adherence to speed and accuracy which humans are just not able to achieve.

1.2 Critical comparison between manual and automation software testing

1.2.1 Automation Testing

Automated testing uses the assistance of tools, scripts and software to perform test cases by repeating predefined manual actions. The biggest draw of automated testing is that it's noticeably easier to run tests quickly, allowing for larger test coverage across environments, and larger code coverage during the testing phase.

One reason for the conversion from manual to automation is the recent shift from Waterfall to Agile development. While Waterfall is a non-iterative and sequential design process, the newer, incremental approach, termed Agile, allows for more flexibility, feedback, and testing throughout the software development process. This is because an Agile strategy caters to changing requirements throughout the development process by having team members integrate their work frequently and communicate more often with customers, business owners, and designers. However, this process of Continuous Integration and Delivery always demands continuous testing. Manual testing no longer had the luxury of a 2 week testing window, but was now facing testing windows of just 2 hours. Successful Agile development teams implement a large amount of test automation throughout the delivery process to make 2 hour testing windows a reality.

(Alex McPeak, 2017)

The object of Automation is to lower the number of test cases and not be run manually so as to remove the requirement of Manual Testing Altogether.

- 70% faster than the manual testing
- More extensive test inclusion of application features
- Dependable in results
- Ensure Consistency
- Saves Time and Cost
- Improves exactness
- Human Intervention is not required while execution
- Re-usable test scripts
- Test Frequently and thoroughly
- More cycle of execution can be accomplished through automation
- Early time to showcase

1.2.2 Manual Testing

Manual testing is the process of checking a software product against functional and non-functional requirements by hand. It is typically conducted by a quality assurance team.

This means that the testers actually run your app on different devices, and use it exactly as your end users would, in order to find any deviations from the original requirements. The team relies on the predefined test cases to check all features and possible use scenarios and find any errors or flaws with the app's functionality, UX, and design

For small scale building efforts, exploratory testing might be sufficient. With this informal methodology, the tester does not pursue any rigorous testing technique, yet rather explores the user interface of the application using as huge numbers of its features as possible, using information picked up in prior tests to instinctively infer extra tests. The success of exploratory manual testing relies intensely on the space expertise of the tester, because an absence of information will prompt incompleteness in testing. One of the key advantages of an informal methodology is to pick up an instinctive insight to how it feels to use the application.

1.2.3 Comparison

1. The automated testing is not replacement of manual testing because all the testing steps may not be automated for example where the change frequency of requirement is high or where the extensive domain knowledge is for testing is required.
2. Test suites needs time to produce valuable results. That is reason upfront cost of automated testing always high.
3. All test executions supposed to be equally consume the resources.
4. All the assumptions are made for a single type of project; even multiple releases of a single project can change the cost of testing

Automated Test might completely remove the need for manual testing, since a PC can perform way more tasks than an average person doing manual testing, it can automatically run test, one such example is CircleCI (Continuous Integration and Delivery) which automatically runs the tests defined by the Software Engineer and run it everytime there are changes made to the source code, this is reducing the human resources required and thereby also reducing the costs. What's more is that the testing tools are becoming more and more intelligent day by day with the advance of AI.

1.3 Justification with stance taken and conclusion

The manual and automated testing is being used in exact proportion. The decision of automating the test cases or not to automate them is always important and critical for the success of automation projects. There is need for research on tradeoff between automated and manual test cases contrast with test case selection, test case reduction, test case prioritization and test case augmentation techniques.

The rationale behind this merger is to identify those already automated test cases to execute manually or some manual test cases in previous test suite need to automate for new objectives like code changes, specification changes, coverage criterion or cost and time issues. This dynamic trade-off between automated and manual test cases may enhance the fault identification capability, risk mitigation and cost and time issues with these techniques.

We conclude the following

1. Cost of testing is considered and indirect costs are not included in this study.

2. This model considers only a single project decision making.

3. Indicating the best possible proportions between two testing types manual and automated.

4. Comparing replacement of one type, production possibility of the type of testing with the other.

5. A single objective testing time considered in this study.

(Trade-off between automated and manual testing: A production possibility curve cost model, 2016)

2 Question 2

Solution to Question 1 Part B

2.1 Introduction to state chart

A State chart diagram depicts a state machine. State machine can be characterized as a machine which characterizes diverse states of an object and these states are constrained by external or internal events.

State chart diagram is one of the five UML diagrams used to show the dynamic idea of a system. They characterize distinctive states of an object amid its lifetime and these states are changed by events. State chart diagrams are valuable to demonstrate the reactive systems. Reactive systems can be characterized as a system that reacts to external or internal events.

State chart diagram portrays the flow of control starting with one state then onto the next state. States are characterized as a condition in which an object exists and it changes when some occasion is activated. The most important reason for State chart diagram is to demonstrate lifetime of an object from creation to end.

State chart diagrams are additionally utilized for forward and reverse engineering of a system. Nonetheless, the principle reason for existing is to demonstrate the reactive system.

Following are the primary motivations behind utilizing State chart diagrams –

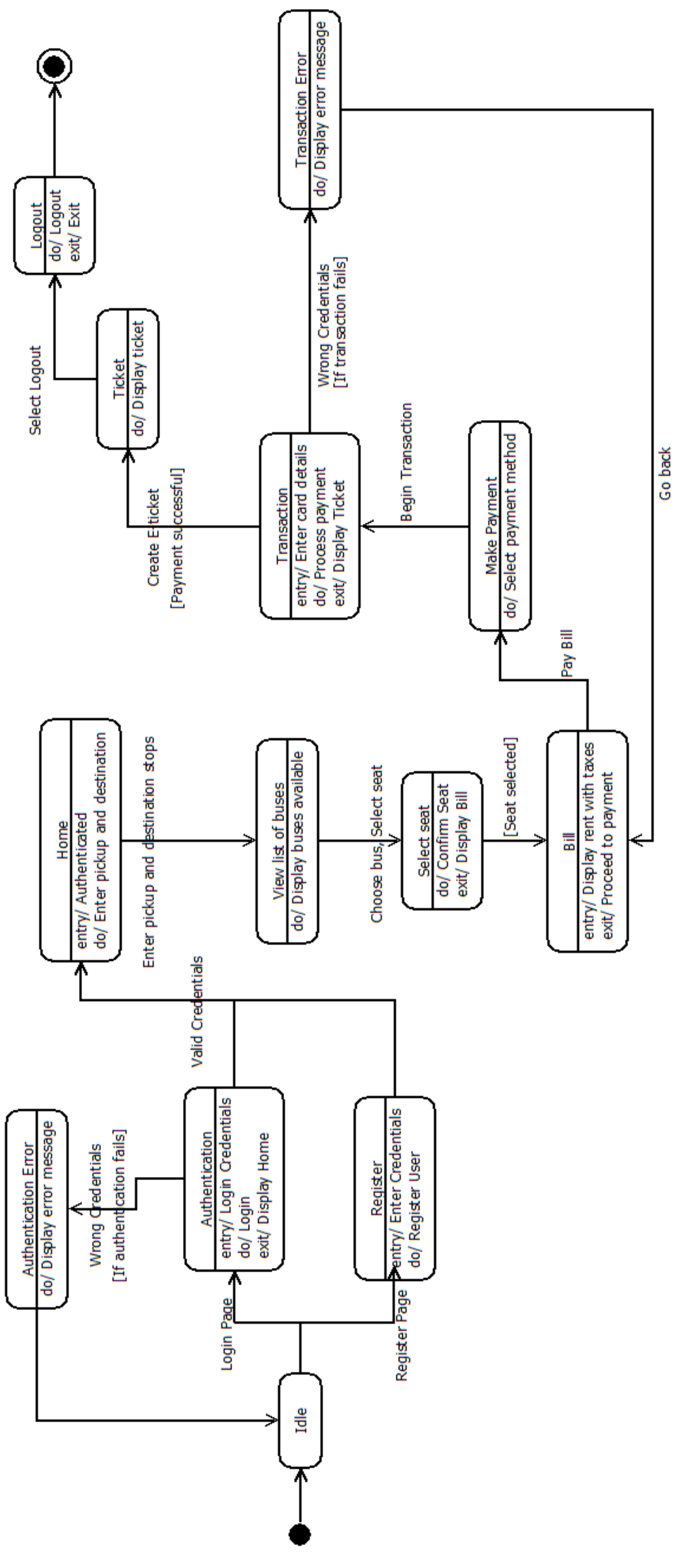
- To demonstrate the dynamic part of a system.
- To demonstrate the existence time of a reactive system.
- To portray distinctive states of an object amid its life time.
- Characterize a state machine to show the states of an object.

(Tutorials Point)

2.2 State chart design

The state chart diagram is based on online bus ticket booking system.

Initially, a user login a software (on the off chance that the user is new, at that point sign in). After login check the seat availability in the following options (if seats are not available at that point go final state). After that, payment for booking tickets. This software allows distinctive sorts of payment, for example, wallets and so forth. Now, e-ticket is displayed.



2.3 Conclusion

State chart diagram is the most valuable strategy in the designing of software.

State diagrams are the ideal way to display object life cycles. State diagrams enable you to portray the behavior of articles during as long as they can remember span. In addition, the distinctive states and state changes as well as occasions causing transitions can be depicted.

In this Context the State Chart Diagram for the Required Bus Management System was designed in Dia keeping in mind the previously designed UML diagrams for the same. The Same Object in different states is depicted in the diagram to show how it transition through the different states of the site, with guards to prevent unauthorized access to some specific sections of the site.

3 Question 3

Solution to Question 2 Part B

3.1 Introduction to Entity-Relationship diagram

An entity-relationship diagram (ERD) is a data modeling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.

The elements of an ERD are:

- Entities
- Relationships
- Attributes

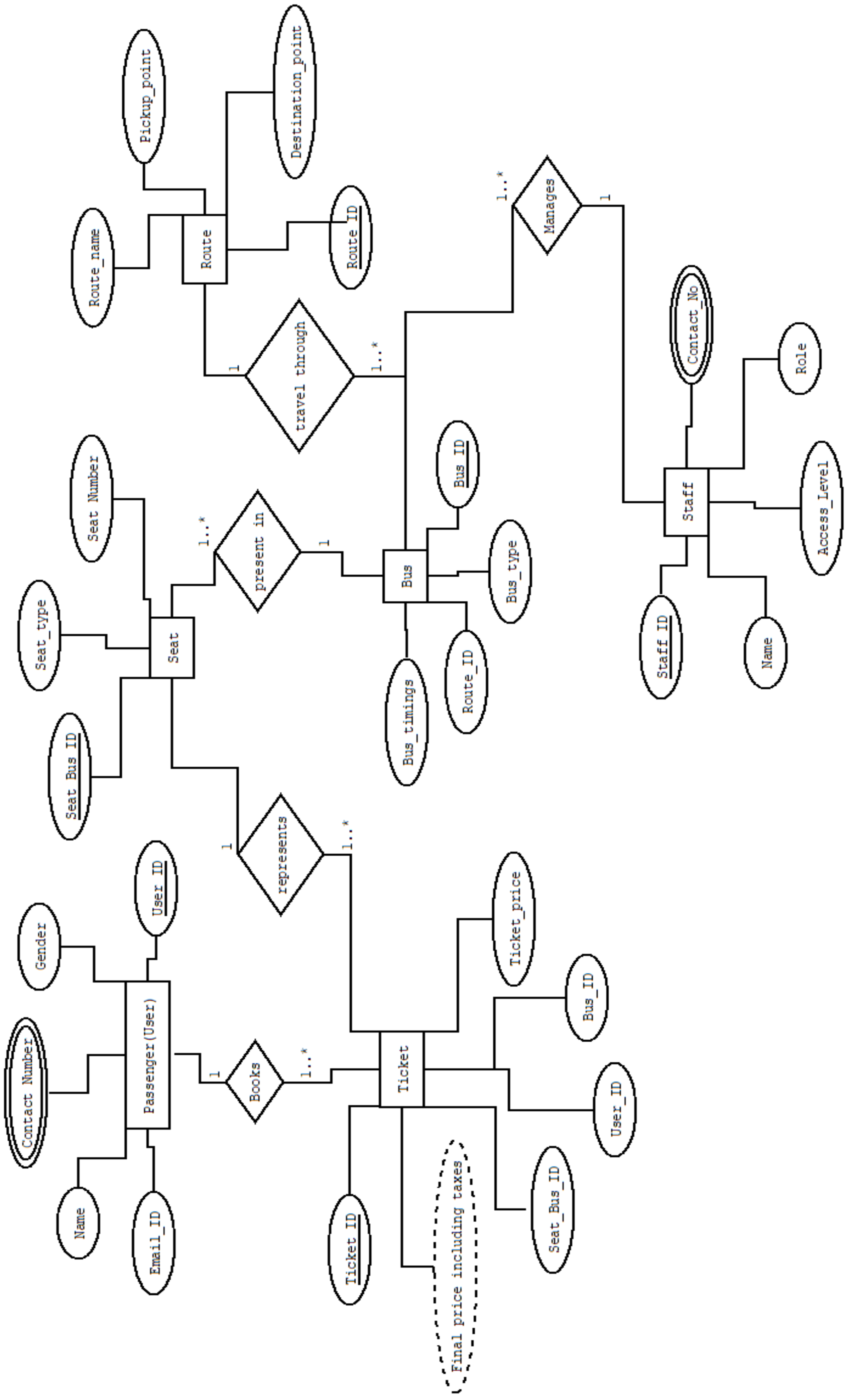
Steps involved in creating an ERD include:

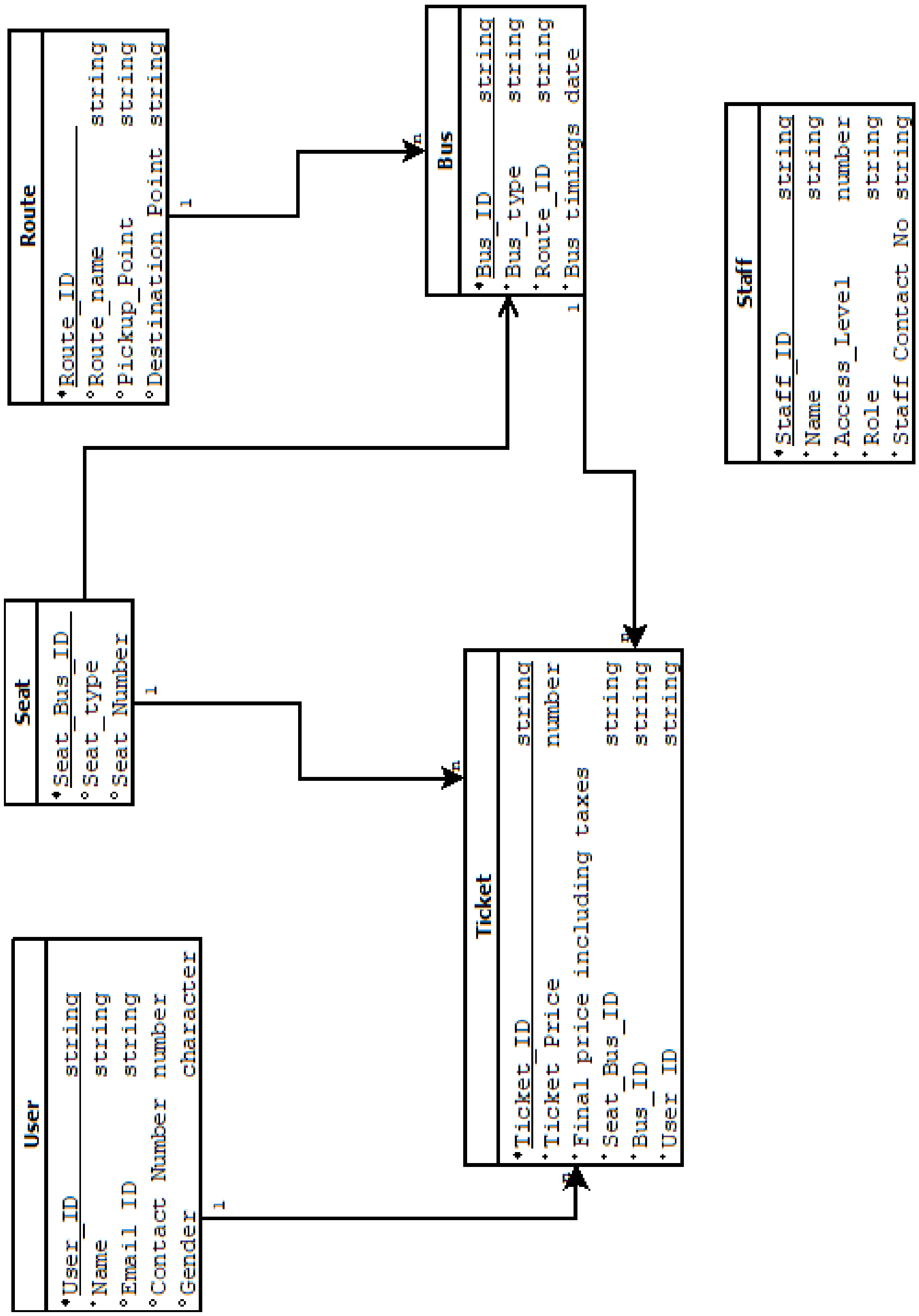
1. Identifying and defining the entities
2. Determining all interactions between the entities
3. Analyzing the nature of interactions/determining the cardinality of the relationships
4. Creating the ERD

An entity– relationship model depicts interrelated things of enthusiasm for a particular domain of learning. A fundamental ER model is made out of entity types and determines relationships that can exist between entities.

In software engineering, an ER model is generally formed to speak to things a business needs to recollect so as to perform business forms. Therefore, the ER model turns into an abstract data model, that characterizes a data or information structure which can be actualized in a database, normally a social database.

3.2 Entity-Relationship design diagrams




Route
Route_ID
Route_name string
Pickup_Point string
Destination_Point string
Bus
Bus_ID string
Bus_type string
Route_ID string
Bus timings date
Staff
Staff_ID string
Name string
Access_Level number
Role string
Staff Contact No string
Seat
Seat_Bus_ID
Seat_type
Seat_Number
Ticket
Ticket_ID string
Ticket Price number
Final price including taxes
Seat_Bus_ID string
Bus_ID string
User ID string
User
User_ID string
Name string
Email ID string
Contact Number number
Gender character
1
n
1
n
1
n
1
n

3.2.1 Example for the Database Table

User

User_ID	Name	Email_ID	Contact_Number	Gender
123456	Satyajit Ghana	satyajig7@gmail.com	7892137665	M

Ticket

Ticket_ID	Ticket_Price	F_Price	S_B_Id	Bus_ID	User_ID
45	120.00	123.23	3	67	123456

Seat

S_B_ID	Seat_Type	Seat_Number
3	Economy	20

Bus

Bus_ID	Bus_Type	Route_ID	Bus_Timings
67	Volvo	23	11AM-8PM

Route

Route_ID	Route_Name	Pickup_Point	Destination_Point
23	Bangalore-Mangalore	Yeshwanthpur	Mangalore

Staff

Staff_ID	Name	Role	Staff_Contact_Number	Access_Level
786734	Some Name	Manger	9837482748	A

3.3 Conclusion

The Entity Relationship Diagram displays the relationships of entity set put away in a database. The ER diagrams are easy change to any data model and it is conceptually straightforward as compare to all different diagrams, for example, state chart diagram, activity diagrams and so on.

Disadvantages of ER diagram are

1. Loss of information content

2. restricted relationship representation

3. No representation of data manipulation

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database.

(Chaitanya Singh, Beginners Book)

Bibliography

1. https://www.tutorialspoint.com/uml/uml_statechart_diagram.htm
2. https://beginnersbook.com/2015/04/e-r-model-in-dbms/
3. https://dzone.com/articles/are-you-confused-between-scripting-testing-and-rec
4. Kazmi, Rafaqat & Ghani, Idris & Mohamad, Radziah & Tariq, M & Bajwa, Imran & Jeong, Seung Ryul. (2016). Trade-off between automated and manual testing: A production possibility curve cost model. 8. 12-27.
5. https://www.altexsoft.com/blog/engineering/striking-a-balance-between-manual-and-automated-testing-when-two-is-better-than-one/
6. https://intellipaat.com/blog/automation-vs-manual-testing/
7. https://crossbrowsertesting.com/crossbrowsertesting/media/pdfs/moving-from-manual-to-automated-testing.pdf
8. https://dzone.com/articles/automated-testing-vs-manual-testing
9. https://crossbrowsertesting.com/blog/continuous-integration/continuous-integration-testing-delivery/