NAME:- TALHA HANDEED

ID:- 14526 (BS(SE))

SECTION:- #(A)

QUESTION NO # 1

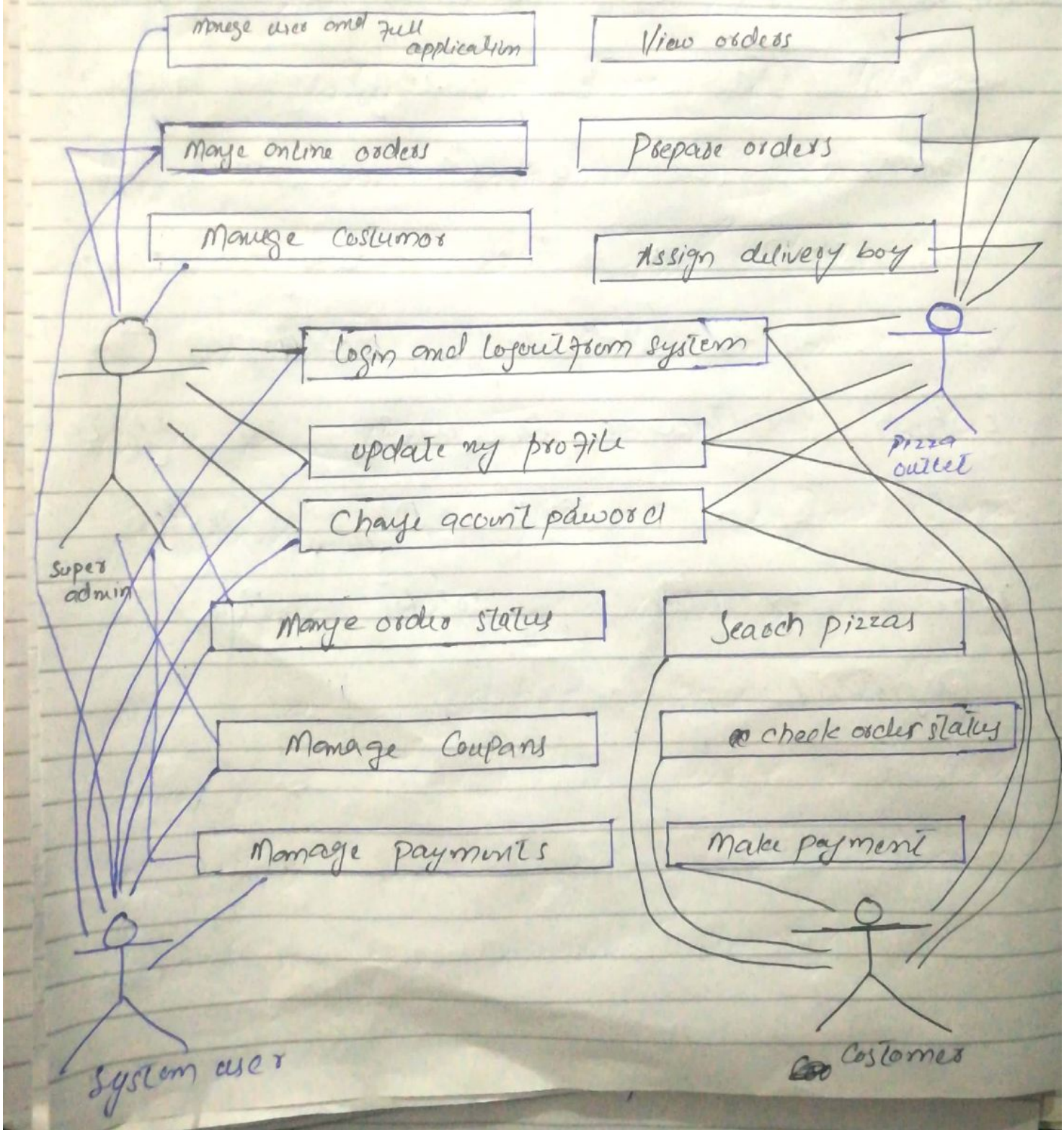
The pizza ordering System allows the users of a web browser to order pizza for home delievery

---

Develop a use case diagram for a use case for placing an order (placeorder). The use case show a relationship to two previously Specified use cases, 'Identify Consumer' which allow a user to register and log in and pay by Credit which model credit card payments.

ANSWER:-

Case Diagram of Pizza Ordering System:



Super Admin Entity:- Use Case of Super Admin are manage pizza, Manage online order, Manage Coupans, Manage payments Manage users and full pizza ordering System operations.

System User Entity:- Use cases of System User are Manage pizzas, Manage online order, Manage Customers, Manage Order status, Manage Coupans, Manage payments.

Pizza Outlets Entity:- Use cases of pizzas outlets are View orders, Prepare ordres Assing delivery boy, Collect payments.

Customers:- Entity:- Use cases of Customers are Search Pizzas Add to card, Make Payments, Check order Status.

QUESTION No # 2

Suggest how an engineer responsible for drawing up a System requirements Specification might keep track of the relationships b/w Functional and Non Functional requirements.

Ans:-

The classic tools is requirements traceability (Sic) matrix. There are even tools dedicated to just tracking requirements. Door is very great that of example. you could fundamentally still use an RTM because in reality its just a spread sheet and numbering Scheme you maintain to track it

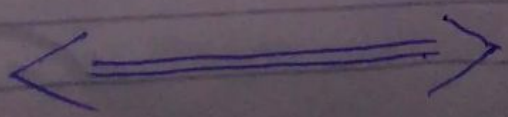
relationship. In agile development the index cards representing uses stories must be tied to requirement you isolated from them. Creating addition work that doesn't provide as much value in terms of testing these stories are indeed to mostly need only a few tests and be realively atomic. what i mean by that is this. you might as well perform the testing before trying to trace. defining "requirements" as rule as possible while still developing test as early as possible. If you are following an agile pradigm. you would develop your

test and tie them directly to each user story. Since these tests answer the question "How do we know that As a user I want to be able to add multiple items to my shopping cart has been implemented correctly. you might have a related tech-story like 'Add the shopping cart library Using this example, hopefully you can imagine requirements and captured through demonstration and testing, both functional and non functional and. Additionally, and hopefully reinforcing these point your requirement are implicitly tested by stakeholder involvement and during the demo-once this feedback loop Successfully and you

you will have fully met the Kissim the requirements are interested to encapsulate.

IMHO large Software System make the hundred of iterations worth of stories difficult to trace back when creating specs. So its best to have tests clearly linked to stories. To date I know of no Software System tat I personally find enjoyable that attempt to make traceability user friendly over multiple iteration. (or long term on a non agile project) Simplified to what I need to do 20% of the time and work well. Worse yet, politics causes problem with Development because

tools regarding traceability are often not Socially engineered to help Solve problems they are often not Socially engineered to help Solve problem they are often engineered to lay blame. There is a huge difference between how these two types of Software operate. Be aware you are better of with tools that make to Solve problems tools) you will know the difference if the Software tries to identify and create process their Control what goes into respoby. As Such Consider fundamentally why you are trying to have thing out and then choose the path approriate for you



QUESTION No # 3:-

To reduce cost and the enviroment impact of commuting your company decides to close a number of offices and to Provide Support for staff to work from home. However the Senior management who introduce the Policy are unaware that Software is developed using agile method which rely on close team working and pair programming. Discus the difficulties that is new policy might cause and how you might get around these problem.

ANSWER:-:-

One of the very best way to give back to the community is to be a good agility - Walk the talk. That itself give so much of good to the community - What I mean by that is below.

1. Learn, get education and keep up to date on the subject.

2 Adopt agile mindset and help the fellow members to adopt the same.

3. Don't be a parrot - Some agile practitioners are just only having, action wise they only teach what is being taught at the classroom. Practical experience is nill - Try to

aside from that

Don't collect certificates, be practical. Some practitioners collect certificates without even understanding the subject matter of behind that certificate. Study just to expand the knowledge.

Give a hand to your fellow colleges. We are not in a competition, there are so much to do. So lets get together and give a helping hand for those who need.

Other ways to give back to the community is to mentor juniors & exam readers, POs or dev teams.

Writing a blog and educate via that

May be use the social mead as a platform. Attend seminars, conferences and help the voluntary work. This is a very common scenario. If you feel that you spend a significant portion of the iteration working production issues then consider doing the following.

1. Track the unplanned work

Work the production issues as usual but track how much time you spending them. One team I've worked with has one product backlog item called "production issues" and they add a task for every issue ~~that~~ they have to work and

include how many hours it took to resolve. Another team I worked with added a product backlog item with a story point for each production issue.

You'll have some real metrics to work with to determine what you need to do.

2. **After an iteration or two, review the unplanned work.**

I'd try to answer the following questions:

- How much time was spent working production issues.
- What would our velocity have been without these issues?

3 How many issues were show stoppers?

* many times I see teams stop everything and work minor issues when either they could have waited or just could have never been done

4. What patterns are there to the production issues? i.e. are most critical issues around a certain feature | system | module | etc?

3. Come up with a plan to reduce the number of production support issues & how to handle them.

Many teams I see teams spend 40 - 50% (or more) of their time

on supporting production. This is a sign that you need to stop building new features and stabilize your current system.

You should always try to account for production issues in some way in the product backlog and dedicate capacity/points. This way it is visible to everyone and it will unable you to inspect & adapt. The reality is, it doesn't affect much because most project cost estimation techniques are woefully inadequate and have failed to deliver on their promise.

Simply put, trying to put a

dollar cost estimate on a project that's going to last more than 6 months is filled with the same biases and failures that putting ~~after~~ effort estimates to such work is — which is precisely the kind of thing that Agile was put in place to replace.

A far more reasonable approach to projecting costs on an Agile project is to fund a set number of resources for a set period of time, and to place checkpoints during the process to assess whether the work as a whole is on target, behind target,

or ahead of target- and then to adjust as. For example; let's say you have three teams of seven development resources running at an average cost of $150,000 per year. That's a total of $ $3,150,000 annually; if you want those teams to work on a project, you would fund that project of $1,575,000 for six months, with checkpoints every month. As those checkpoints demonstrate progress, you adjust accordingly.

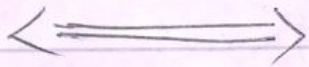
Over time, you'll collect data to understand what projects seems to fall within your expectations, and

which do not - You can then use retrospection to identify the common traits of those which do and attempt to apply those traits to the ones that do not.

Basically, under Agile you're not funding **projects** but rather **products** — on going efforts without any specific end date but with business goals and check points to ensure that they are delivering the expected value.

QUESTION NO # 4

Discover difficulty / ambiguities or omisions in the following statement of requirement for part of a ticket issuing - System.



ANSWER:-

Discoved ambiguities or omision in the following statement of the required for part of ticket-issuing System.

An automated ticket machines sells rail tickets users select their ticket destination and input a credit card and a personal identification number. The rail ticket is issued and their credit card account charged

when the user presses the start button a menu display of potential destination is activated along with a message to the user to select a destination has been selected. The ticket price is displayed and customer are asked to input their credit card is validity is checked and the user is then asked to input his or her personal identifier (PIN). When the credit transaction has been validated the ticket is issued.

The ticket issuing system does not offer many of the service to facilities the purchase of ticket.

A number of ambiguities and omision have been found from the given Scenario.

The Scenario given not clear enough and looks like quite uncertain about the instruction to purchase a ticket. Even the payment procedures do not offer the user to choose the type of payment. The System does not allow costomuma buy more then one ticket for the same destination at a time. it must prompt the user if the ticket can be purchase one at a time. Costomer is more likely to input incorrect destination and the System should allow the user to cancel

a request and purchase another ticket. After having input the destination System will ask the Costomer to pay and would ask about which type of payment the user prefer. in this case the System does not indicate as to how to respond if an invalid card is input. It must be able to respond and promt the user if they try to put their card before Selecting a destination. Do the Costomer need to press the start button they wish to buy another ticket to a different destination. Or it allows ~~the~~ Customers with the choice of Selecting other distination between the stations where the machines is located.

Ticket System is going to allow input their destination through a touch screen or keyboard. The exiting System does not allow the user to view the ticket prices accordingly, as Customer need to know the amount that they will be charged. it does not show the train departure and arrival times. and weather the Customer wished to buy ticket for a specific train it must allowed Customer to choose thier seats.

QUESTION NO # 5:-

Using your knowledge of how ATM is used, develope a set of use Cases that could Serve a basis for understanding the requirements for an ATM System.

ANS:-

An automated teller machines (ATM) or the automatic banking machines (ABM) is a banking Subsystem (subject) that provide bank customers with acess to financial transiction in a public Space without the need for a cashier clerk, or bank teller

Customer (actors) uses bank (ATM) to check balances of his, her bank account, Deposit Funds, withdraw cash

and/or Transfer Funds (use cases) (ATM) Tecnicians provide Maintenance and Repairs. All these use cases also involve Bank actor whether it is related to customers transaction or to the (ATM) servicing.