Item Management System : IMS

By: Ben Irwin

GitHub: https://github.com/BenI-QA/ims_project

Outline:

Plan

Project Structure/WorkFlow

UML/ ERD Structure

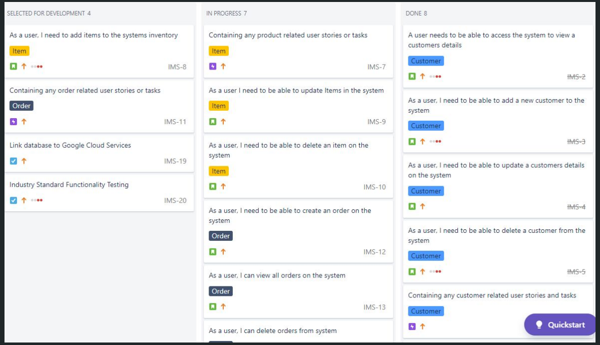
Risk Assessment

Live Demonstration

Identify what went well and improvements

Identifying what i have completed and what was left behind

Introduction



SELECTED FOR DEVELOPMENT 4

As a user, I need to add items to the systems inventory
Item
IMS-8

Containing any order related user stories or tasks
Order
IMS-11

Link database to Google Cloud Services
IMS-19

Industry Standard Functionality Testing
IMS-20

IN PROGRESS 7

Containing any product related user stories or tasks
Item
IMS-7

As a user I need to be able to update Items in the system
Item
IMS-9

As a user, I need to be able to delete an item on the system
Item
IMS-10

As a user, I need to be able to create an order on the system
Order
IMS-12

As a user, I can view all orders on the system
Order
IMS-13

As a user, I can delete orders from system

DONE 8

A user needs to be able to access the system to view a customers details
Customer
~~IMS-2~~

As a user, I need to be able to add a new customer to the system
Customer
~~IMS-3~~

As a user, I need to be able to update a customers details on the system
Customer
~~IMS-4~~

As a user, I need to be able to delete a customer from the system
Customer
~~IMS-5~~

Containing any customer related user stories and tasks
Customer

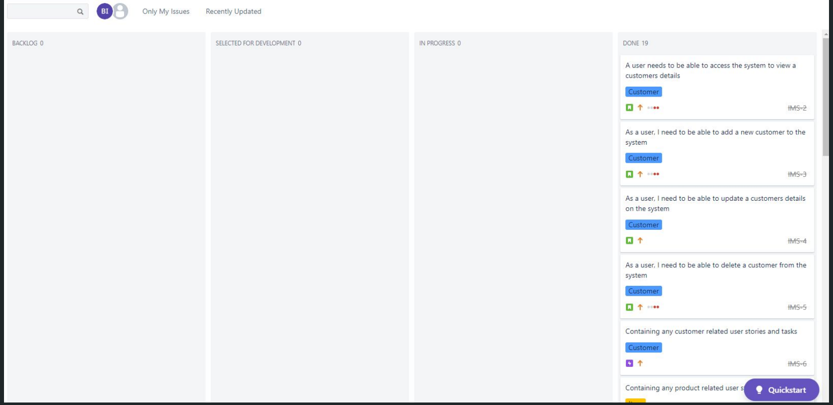
How did I approach this specification?

-Planned out on Jera how I would outline each project using Kanban

-Prioritised tasks based on priority, story points

-Linked each object in project through Epics

-Identified small tasks which improved project



BACKLOG 0

SELECTED FOR DEVELOPMENT 0

IN PROGRESS 0

DONE 19

A user needs to be able to access the system to view a customers details
Customer
~~IMS-2~~

As a user, I need to be able to add a new customer to the system
Customer
~~IMS-3~~

As a user, I need to be able to update a customers details on the system
Customer
~~IMS-4~~

As a user, I need to be able to delete a customer from the system
Customer
~~IMS-5~~

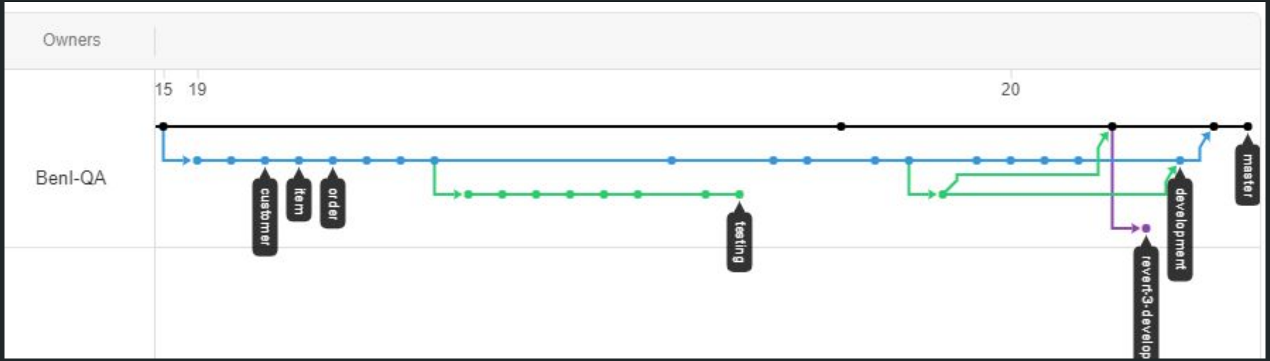
Containing any customer related user stories and tasks
Customer
~~IMS-6~~

Containing any product related user s

Project Management : Workflow

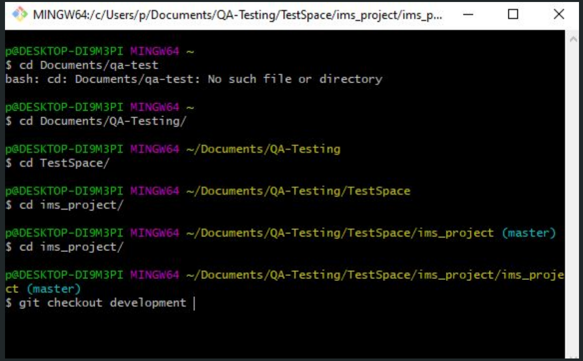
Using Git:

Used git to plan out the different features of my project separately so each stage of the development is isolated and it was fairly easy to revert to previous versions of my work.



Using Git Bash:

Using git bash to move between branch, merge and push my work back to github


MINGW64:/c/Users/p/Documents/QA-Testing/TestSpace/ims_project/ims_p...

```
p@DESKTOP-DI9M3PI MINGW64 ~
$ cd Documents/qa-test
bash: cd: Documents/qa-test: No such file or directory

p@DESKTOP-DI9M3PI MINGW64 ~
$ cd Documents/QA-Testing/

p@DESKTOP-DI9M3PI MINGW64 ~/Documents/QA-Testing
$ cd TestSpace/

p@DESKTOP-DI9M3PI MINGW64 ~/Documents/QA-Testing/TestSpace
$ cd ims_project/

p@DESKTOP-DI9M3PI MINGW64 ~/Documents/QA-Testing/TestSpace/ims_project (master)
$ cd ims_project/

p@DESKTOP-DI9M3PI MINGW64 ~/Documents/QA-Testing/TestSpace/ims_project/ims_proje
ct (master)
$ git checkout development |
```

Consultant Journey

What technologies were learnt from this project?

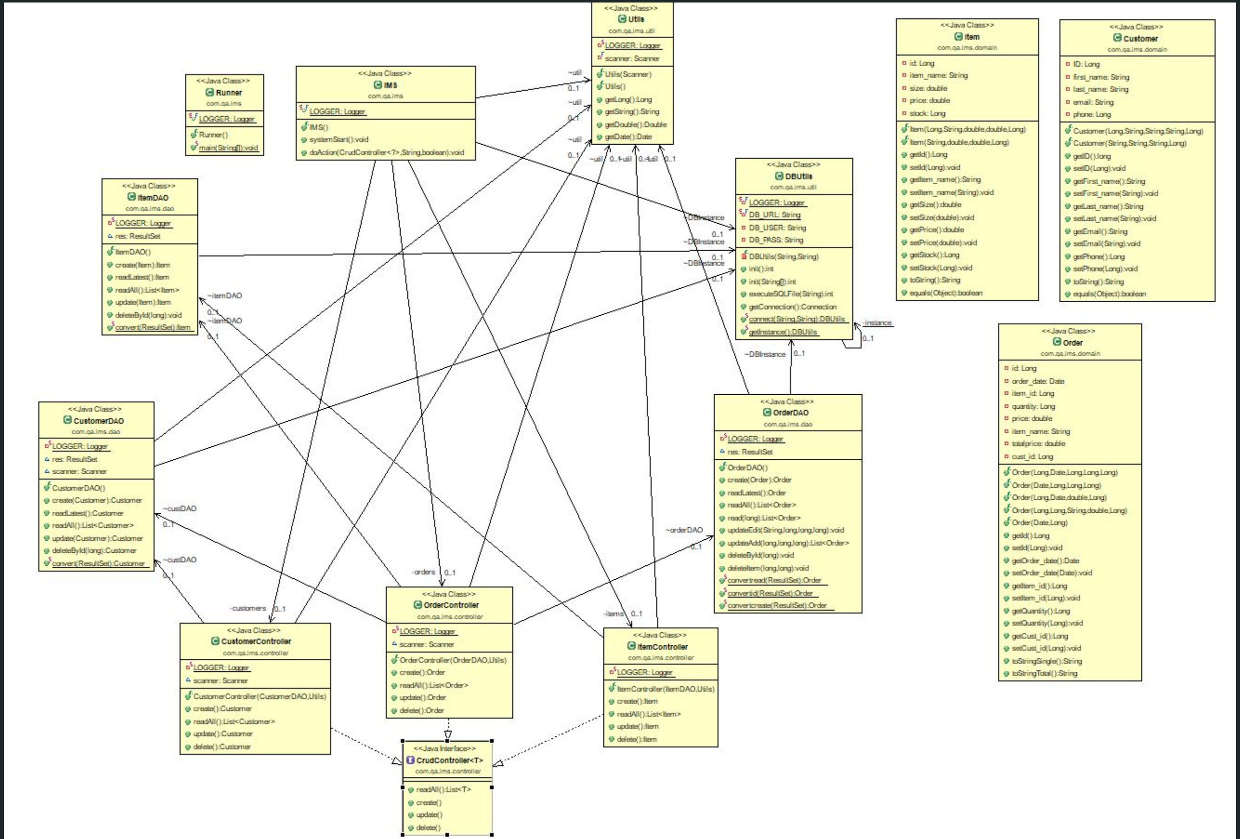
-Intermediate Java design and techniques

-SQL interaction with java

-JUnit and Mockito

UML Diagram of Program

Each Relatable class connecting wherever it is used



<<Java Class>>
Runner
com.qa.ims
LOGGER: Logger
Runner()
main(String[]):void

<<Java Class>>
IMS
com.qa.ims
LOGGER: Logger
IMS()
systemStart():void
doAction(CrudController<?>,String,boolean):void

<<Java Class>>
Utils
com.qa.ims.util
LOGGER: Logger
scanner: Scanner
Utils(Scanner)
Utils()
getLong():Long
getString():String
getDouble():Double
getDate():Date

<<Java Class>>
Item
com.qa.ims.domain
id: Long
item_name: String
size: double
price: double
stock: Long
Item(Long,String,double,double,Long)
Item(String,double,double,Long)
getId():Long
setId(Long):void
getItem_name():String
setItem_name(String):void
getSize():double
setSize(double):void
getPrice():double
setPrice(double):void
getStock():Long
setStock(Long):void
toString():String
equals(Object):boolean

<<Java Class>>
Customer
com.qa.ims.domain
ID: Long
first_name: String
last_name: String
email: String
phone: Long
Customer(Long,String,String,String,Long)
Customer(String,String,String,Long)
getID():long
setID(Long):void
getFirst_name():String
setFirst_name(String):void
getLast_name():String
setLast_name(String):void
getEmail():String
setEmail(String):void
getPhone():Long
setPhone(Long):void
toString():String
equals(Object):boolean

<<Java Class>>
ItemDAO
com.qa.ims.dao
LOGGER: Logger
res: ResultSet
ItemDAO()
create(Item):Item
readLatest():Item
readAll():List<Item>
update(Item):Item
deleteById(long):void
convert(ResultSet):Item

<<Java Class>>
DBUtils
com.qa.ims.util
LOGGER: Logger
DB_URL: String
DB_USER: String
DB_PASS: String
DBUtils(String,String)
init():int
init(String[]):int
executeSQLFile(String):int
getConnection():Connection
connect(String,String):DBUtils
getInstance():DBUtils

<<Java Class>>
Order
com.qa.ims.domain
id: Long
order_date: Date
item_id: Long
quantity: Long
price: double
item_name: String
totalprice: double
cust_id: Long
Order(Long,Date,Long,Long,Long)
Order(Date,Long,Long,Long)
Order(Long,Date,double,Long)
Order(Long,Long,String,double,Long)
Order(Date,Long)
getId():Long
setId(Long):void
getOrder_date():Date
setOrder_date(Date):void
getItem_id():Long
setItem_id(Long):void
getQuantity():Long
setQuantity(Long):void
getCust_id():Long
setCust_id(Long):void
toStringSingle():String
toStringTotal():String

<<Java Class>>
CustomerDAO
com.qa.ims.dao
LOGGER: Logger
res: ResultSet
scanner: Scanner
CustomerDAO()
create(Customer):Customer
readLatest():Customer
readAll():List<Customer>
update(Customer):Customer
deleteById(long):Customer
convert(ResultSet):Customer

<<Java Class>>
OrderDAO
com.qa.ims.dao
LOGGER: Logger
res: ResultSet
OrderDAO()
create(Order):Order
readLatest():Order
readAll():List<Order>
read(long):List<Order>
updateEdit(String,long,long,long):void
updateAdd(long,long,long):List<Order>
deleteById(long):void
deleteItem(long,long):void
convertread(ResultSet):Order
convertid(ResultSet):Order
convertcreate(ResultSet):Order

<<Java Class>>
CustomerController
com.qa.ims.controller
LOGGER: Logger
scanner: Scanner
CustomerController(CustomerDAO,Utils)
create():Customer
readAll():List<Customer>
update():Customer
delete():Customer

<<Java Class>>
OrderController
com.qa.ims.controller
LOGGER: Logger
scanner: Scanner
OrderController(OrderDAO,Utils)
create():Order
readAll():List<Order>
update():Order
delete():Order

<<Java Class>>
ItemController
com.qa.ims.controller
LOGGER: Logger
ItemController(ItemDAO,Utils)
create():Item
readAll():List<Item>
update():Item
delete():Item

<<Java Interface>>
CrudController<T>
com.qa.ims.controller
readAll():List<T>
create()
update()
delete()

Entity Relationship Diagram

Relationships

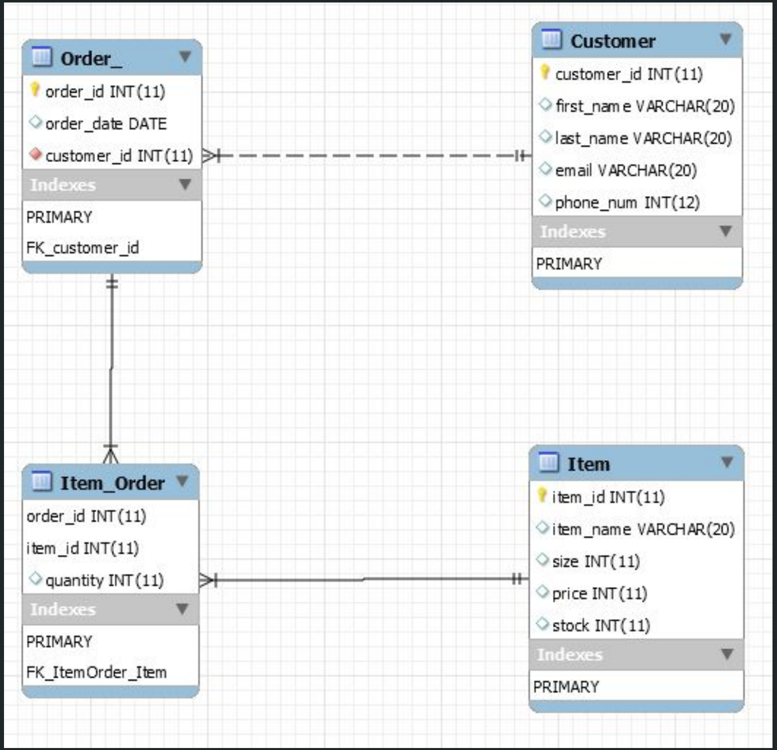
Customer to Order: one-to-many

Item to item_order:one-to-many

Order to item_order: one-to-many

Primary key for composition table:

(order_id, item_id)



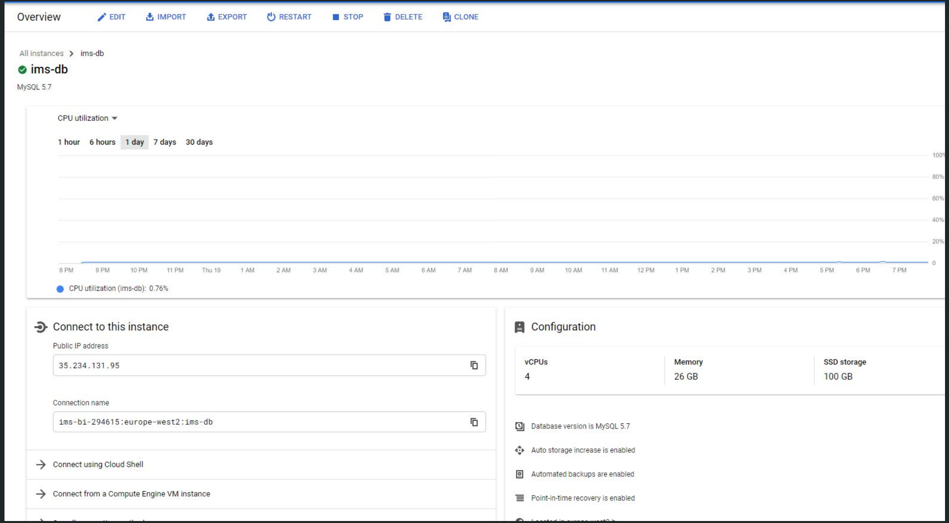
Google Cloud SQL Database

Benefits

- More Storage for Records
- SQL_Schema file can run from any machine

Drawbacks

- Currently has an open firewall



ims-db

MySQL 5.7

CPU utilization (ims-db): 0.76%

Connect to this instance

Public IP address: 35.234.131.95

Connection name: ims-bi-294615:europe-west2:ims-db

Configuration

vCPUs	Memory	SSD storage
4	26 GB	100 GB

Database version is MySQL 5.7

Auto storage increase is enabled

Automated backups are enabled

Point-in-time recovery is enabled

Risk Assessment

Initial thoughts on what might be a risk

Final thoughts on Risk Assessment and what issues were encountered

Likelihood

Rare	Unlikely	Possible	Likely	Certain
1	2	3	4	5

Impact

Negligible	Minor	Moderate	Major	Extreme
1	2	3	4	5

Risk Level

Low	Moderate	High	Extreme
1 – 3	4 – 6	7 – 8	9 – 10

Risk Register 1: Initial Scores & Actions Taken

Ref	Risk Description	Cause	Risk Event	Likelihood (1-5)	Impact (1-5)	Risk rating	Action
1	Lack of Time	Improper time management	Not completing project/task	1	4	4	Plan daily and weekly tasks, set deadlines
2	Self-Management	Burnout	Not being able to complete work, physical illness	3	3	9	Take regular breaks, set realistic goals, ask for help when needed
3	Worldwide Disruption	Covid-19 Outbreak	Illness, upheaval, global infrastructure strains	3	3	9	Social distancing, wash hands regularly
4	Computer Crash	Poor Hardware	Loss of work	3	4	12	Keep multiple backups of work
5	Internet loss	Remote location	Unable to connect to cloud	1	2	2	Save database to local machine & change variables

Risk Register 2: Final Scores & Actions Taken

Ref	Risk Description	Likelihood (1-5)	Impact (1-5)	Risk Rating	Analysis	Action
1	Lack of Time	1	4	4	Planning out each stage of the project unless an unexpected error occurs	Plan daily and weekly tasks, set deadlines
2	Self-Management	3	3	9	I could of management my time a lot better as I came across mainly compiler issues with maven towards the end	Take regular breaks, set realistic goals, ask for help when needed
3	Worldwide Disruption	3	3	9	Risk of going outside could of impacted the development of the project if i was sick	Social distancing, wash hands regularly
4	Computer Crash	3	4	12	Computer crashes multiple times during development, however saving worked regularly was the right thing to do	Keep multiple backups of work
5	Internet loss	1	4	4	Due to the home which I'm working the internet was fairly bad and was difficult to upload	Save database to local machine & change variables
6	Lack of understanding	3	3	2	Came across issues such as mockito and maven compiler which held me back	Paced myself and prioritised urgent tasks first

Live Demonstration

What did you complete?

Completed

Customer Epic

Item Epic

Order Epic

JUnit/Mockito Testing

Left Behind

-OrderUpdate function having issues

-10% more testing for JUnit and Mockito

-running using a fat .jar file

What went well and What I could Improve

-What went well?

-Built confidence in navigating my own code and open source code

-Was quick adapting to changes and error monitoring

-Improvements

Order functionality could be optimised

Could of made more use of more java design principles such as adapters

Conclusion