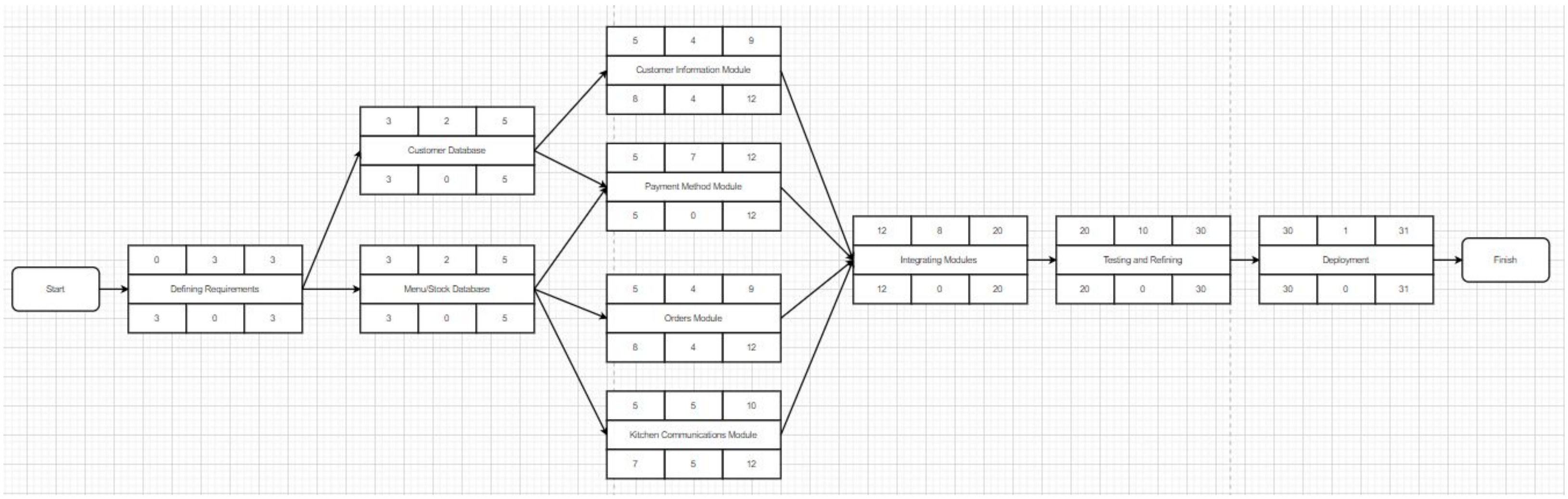
SOFE 3490 Lab 4:
Software Project Management

Wednesday March 25th, 2020

Meghana Govind	100620126
Sheila Kuria	100618783
Tasfia Alam	100584647

Activity Diagram:



Tasks and Risk Management:

Task	**Risk**	**Countermeasure**
Creating Customer database: Developers will develop a well encrypted database	This database will store customer information, from their names, mailing address, past orders and the amount of points the customers have accumulated from ordering from our restaurant. Biggest concern and risk in creating a database that stores our customers information is security or data leaks.	Seperate the customer database from the web server. Encrypt stored files as well as database backups. This might require extra resources like time, developers and money to ensure we have top tire encapsulation. Allocating our resources properly especially to security is a priority
Module to retrieve customer's information: Module created to retriver customers information	Customers may not provide the database with accurate information. Example is giving the database a street address for Toronto but a postal code for Kingston Ontario.	Integrate the database with API maps that checks if the postal code and street addresses complement each other.
Integrating all modules: Brining all modules together	Some tasks may not be done on time to integrate the system.	Try to finish the project a few weeks before the deadline thus there is enough legroom if any errors or concerns occur.
Testing: Testing the system with all the parts	Testing may take up more time than expected, especially if we want an excellent product.	Shareholders may be willing to take the risk of the product being deployed at a later time, it's more costly to deploy a product that is not properly tested as opposed to releasing a product that lives up to its name and the restaurant's reputation.
Deployment: Make the system available to all	Unsatisfied customers.	It's disheartening to release a product that will get bad reviews, but with excessive work and testing it shouldn't be a huge issue. Stakeholders should consider what users are saying about the product after release, to ensure that upgrades and future releases consider our customer's voice.

<u>Defining Requirements</u>: Stakeholders come together to define the requirements of the system	All requirements of the system need to be defined precisely and thoroughly, thus a major risk associated with this task is if certain requirements are missing or certain stakeholders do not define requirements.	A countermeasure that can be taken to ensure all stakeholders are accounted for and all requirements are thoroughly and precisely defined is to do proper requirements analysis using methods such as use cases, class-based methods, and domain analysis
<u>Menu/Stock Database</u>: Creating the database of the menu items and the stock of ingredients	If the menu does not update according to the stock, then customers and kitchen staff will not be informed of the availability of items, thus disrupting restaurant operations.	Incrementally develop modules to ensure that interfaces are properly working and dependency is minimized.
<u>Payment Module</u>: Creating a module that integrates a POS system to process transactions	An external Point-Of-Sale System will be used, such as a Moneris machine, to accept and process popular forms of payments. However, if the system has compatibility issues with the external device, then issues with transactions arise.	Choose a POS system with formal specifications and have quality controls (such as external QA testers) for the system with the POS devices.
<u>Kitchen Communications Module</u>: Creating a module that will receive the table orders, update the menu/stock	The Kitchen Communication Modules do not have the proper user interface or the proper interface between the orders and kitchen.	Have kitchen management involved with the testing and requirement specifying for this component. Require a test analysis for this component.
<u>Orders Module</u>: Creating a module that will create an order for a corresponding table and for each customer	The ordering module does not properly execute its functions, such as storing and deleting orders to table numbers, saving and updating orders, sending orders to the kitchen.	Prototype the order module and use formal specification methods to design and build the ordering module.

Resources Allocation:

Requirements Gathering	
Defining Requirements	George, Jess, June, **Tim (Project Manager)**
Software Architecture Team	
Menu/Stock DB	George
Customer DB	Jess
Development Team	
Payment Module	June
Customer Information Module	Tim, William
Communication Module	Joe, Sarah
Order Module	William
Integrate Modules	June, George
Quality Analysis Team	
Testing and Refining System	Jake, April, Tim
Deployment	Tim