Risk Identification

* Change in user requirements
* The project takes longer than previously thought
* Technologies don’t work together the way they should
* Project team members do not know the technologies well enough
* The system is not able to handle the program
* Corrupt data
* Wrong inputs
* Negative stock

Risk Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| Threat | Probability | Control | Feasibility |
| Change in user requirements | low | Make sure that the requirements are understood and build modularly | High cost if requirements not met and major refactoring must be done |
| Timeframe | Medium | Study and research each part to make sure that it is implemented and clear deadlines | This can be done relatively easy as we are still learning and will be researching as we go |
| Technologies not compatible | High | Test the code and technologies thoroughly and make sure that documentation of components used is understood | This is a feasible action as we will be learning in this project how to build the app. |
| Lack of Knowledge | High | Find reusable resources to educate ourselves on the technology | This will be done during the project to educate ourselves and may be time consuming |
| System can’t handle program | Medium | Test thoroughly through test servers and databases before deploying | This is reasonable, but may need more education on deployment before test can be done efficiently |
| Corrupt data | Medium | Data transfer between the program and database must be thoroughly tested and checks placed to prevent | More difficult to implement, but structure and coding constructs can be used to accomplish |
| Wrong Inputs | High | Validate user input before running | This should be clear and transparent and let the user reinput data |
| Negative Stock | Low | Database rules to not allow negatives and validation to check user input and limit their input | Easy to implement through coding constructs and database rules |