Atech Computers

Project Closeout Report

Table of Contents

[1. Project management 4](#_Toc452224302)

[**1.1 Personnel** 4](#_Toc452224303)

[1.1.1 Background 4](#_Toc452224304)

[1.1.2 Skill Set 5](#_Toc452224305)

[1.1.3 Working Time 5](#_Toc452224306)

[**1.2 Planning** 6](#_Toc452224307)

[1.2.1 Scope Control 6](#_Toc452224308)

[1.2.2 Methodology 7](#_Toc452224309)

[1.2.3 Time Management 8](#_Toc452224310)

[1.2.4 Responsibility Assignment 9](#_Toc452224311)

[**1.3 Design** 10](#_Toc452224312)

[1.3.1 Interaction Design 10](#_Toc452224313)

[1.3.2 Page Style Design 11](#_Toc452224314)

[1.3.3 Database Design 12](#_Toc452224315)

[1.3.4 Problems 12](#_Toc452224316)

[**1.4 Communication** 13](#_Toc452224317)

[1.4.1 Meetings 13](#_Toc452224318)

[1.4.2 Information Sharing 14](#_Toc452224319)

[1.4.3 Instructions 14](#_Toc452224320)

[**1.5 Monitoring and Controlling** 14](#_Toc452224321)

[1.5.1 Progress Monitoring 14](#_Toc452224322)

[1.5.2 Quality Control 15](#_Toc452224323)

[1.5.3 Change Control 16](#_Toc452224324)

[1.5.4 Audit 16](#_Toc452224325)

[**1.6 Things That Happened Unexpectedly** 17](#_Toc452224326)

[1.6.1 A Teammate Joined 17](#_Toc452224327)

[1.6.2 Change of Framework 17](#_Toc452224328)

[1.6.3 Reaction Version Migration 18](#_Toc452224329)

[2. New Technologies 19](#_Toc452224330)

[**2.1 Meteor** 19](#_Toc452224331)

[**2.2 MongoDB** 19](#_Toc452224332)

[**2.3 Docker** 20](#_Toc452224333)

[3. General Recommendations for Future Group 21](#_Toc452224334)

# Project management

**1.1 Personnel**

The first trouble of the team came when trying to form a group. It happened to the team leader that project students available for this trimester was quite limited. Only one local student from BIT was to do the project, and 90% of international students were Indians who knew each other and grouped themselves well. There was once a time that he decided to do the project himself.

Weltec project encourages practice of project management and team working. A group of only one person is not recommended by the coordinator. The team leader posted on the forum to recruit more members but no one answered and he was still alone after the academic commencement. However, right after he finished the proposal and was ready to proceed to the next phase, an unexpected new member was introduced to the team. The two members with different skill sets and project background recognitions started their work. Cooperation between such two guys was really a challenge to them. How they would eliminate the misunderstanding and perform with the highest proficiency determined the the success of the project. We knew exactly it was what the coordinator was looking forward to. He intentionally made us into such situations and saw us find a way out.

**1.1.1 Background**

We come from different countries, have grown up under different cultures and believe in different life philosophies. But we do not think that kind of difference of views of world has ever affected our cooperation negatively. Instead, we have different expectations of what to gain from the project. The team leader worked for IT companies for a couple of years, and he know how IT projects are running, the pressure from outside, the methodology, and things like that. He and the adviser are both hoping him to get a quite high mark from the course. But the other member owns a different concept of IT projects. At the beginning, he was mentally against what were planned for management, like meetings, agendas, documents. It made him reluctant to finish the work assigned to him.

**1.1.2 Skill Set**

Actually both of us lack the skills exactly suitable for this project. The team leader was once an Android developer, with only no more than half years’ Java Struts experience. The other is even a networking student rather than a programmer. None of us had the skill to build a web server from the beginning. We know that a lot of techniques are involved on the way to a successful web server and it requires years of experience to achieve that. But the fact is we have to complete such kind of a system within three months, let alone more than one month’s research time.

There was no way out but to get the ball rolling at the moment. Originally, the leader already had some idea from the client and spent about two months before the trimester started researching the possibility to utilize Angular plus Strongloop as the project framework. Although we did not use that combination later in the project, it helped him a lot get familiar in advance with the JavaScript language and the best practice in industry. It guided him step by step to the ReactionCommerce platform which was employed as the starting point of the project.

During the research period, we came across many fresh new concepts, like Meteor, MongoDB, ECMAScript 2015, Node.js, AWS, and Docker, which are quite important and popular in JS-based application development field. We are still in studying stage of those concepts but we feel optimistic to continue similar projects in the future.

**1.1.3 Working Time**

We are supposed to spend 435 hours for this project before the academic deadline, and according to our progress report, we will certainly achieve that goal.

But still we had problem. Vineet has two jobs, and his hours available for this project are quite different from normal students. The team working time is only possible from 2pm to 5pm on weekdays. Considering this, we set up this time mandatory for working together. It turns out that this decision is pretty wise and we gained much benefit from it, although it was not kept strictly. We have heard from other teams that their members met with each other only in team meetings twice a week. It was hard for them to proceed smoothly and they often got stuck because someone suddenly disappeared for a short time exactly when some information had to be synchronized. In this aspect, we are proud of ourselves to have continuous face-to-face communication.

**1.2 Planning**

**1.2.1 Scope Control**

Scope control is planning, performing, and managing how, what to be done during the project time. As building a website from the beginning is quite a large amount of work, and expectations from the client was too general and ambitious, it was easy at at time to get requirements out of control. We knew our restriction of skills, and we were aware that if we were to succeed, we must think up ideas to get it over.

Generally, two solutions were in front of us. First, convincing the client to lower his expectations and listing his requirements in priorities. Only the one with the highest priority was put into plan first, and let the project evolve along with more functionalities added. Even if the academic deadline comes, the project can still go on until it is mature enough to go live. Second, picking up a current online shopping open source project and customizing it. It seems like a much quicker path to a runnable project but it apparently needs more research and rework. The client was unwilling to accept this plan because he had the opinion that those open source solutions were not fully-customized and may be unsafe for an e-commerce.

Finally we adopted them both. While we chose ReactionCommerce as the base platform, the development began by adding functionalities that the client thought were the most identifying ones from other current websites. On top of the list, he mentioned, was importing supplier csv product lists and updating product information each time the suppliers update their database. There were many small technical difficulties on the way, so our plan was to conquer them all and achieve this goal.

The last two phases were designed to be adaptive phases which was proved later to be very useful. In those phases it was flexible enough to add more functionalities or do anything else necessary. Also, they can be considered as buffer time to manage the risk when failing to finish the plan on time. It was our secret of surviving the torture of project.

**1.2.2 Methodology**

Methodology is the philosophy of proceeding and getting the most out of the project. Several models were taught in the textbooks, but when it comes to a certain type of project, it is possible that only one of them is the most suitable.

Waterfall model is pretty old, in which all the processes are arranged in a linear manner. The nodes of the chain run one after one precisely, any change is not allowed to break the original design.

For most general purpose projects, like web applications and mobile applications, waterfall model does not apply well. Because changes often occur during the development period, and schedules are adjusted in a timely manner to accommodate those changes. Furthermore, requirements of such projects are usually not certain at the beginning, but waterfall model demands that all the designs are fixed in the entire software development life cycle.

Therefore, our best choice is the one among the life cycle models that provide flexible processes. Options are incremental and adaptive models, as well as the agile method. Many practices are involved under the agile topic, such as daily stand-up meetings and pair-programming. We cannot carry out those practices in the case that we do not stay with each other all day and we only have one programmer.

From the scope control section above, it is clear that we arrange the requirements into fixed and adaptive phases. The workable website is released version by version as defined by the incremental model. At the end of this stage, importing supplier product list should be available. For the rest of the time, we can plan any work if needed.

This design of methodology helps us a lot. As will be described later, we missed our incremental due day. The buffer time is useful for us to finish technical work. And we leave the last phase for writing reports. The methodology is the most important factor that led us to succeed.

**1.2.3 Time Management**

1.2.3.1 Project Schedule Model

We use Microsoft Project 2013 to create the schedule model. It is a good tool to manage tasks, milestones and particularly it demonstrates the planned evolving processes of the project.

But according to our experience from the project, except for the person who produces this plan, no one else actually cares about what is in it. They even never opened this file once, and of course they never update their progress into the plan, although they have been told many times. It makes statistics of man power performance very hard to be precise.

The reason may be that we have written the tasks of each member in weekly progress report. However, the project manager is definitely happy to see that his team know what is going on and get the idea of the whole picture rather than just doing what they are told to do for this week.

* + - 1. Scheduling Methodology

Scheduling methodology is confirmed based on our SDLC (Software Development Life Cycle) model. As mentioned in 1.2.2, we adopt the mix of incremental model (for fixed requirements) and adaptive model (for flexible planning). Therefore, our scheduling strategy borrows the ideas from conventions of versions or cycles. We split the project into six phases (iterations in agile method): four weeks for research, and two weeks each for the rest. This practice is learned from pervious project experience in China when the agile concept became popular and internet companies were competing with each other to put it into practice. The purpose of this strategy is to release products and get feedback quickly enough to respond the ever changing world. But it is only the beginning of the project, so the “release” is not that obviously important. The main idea of doing so is to split a big project into small periods with manageable goals. It is a good way to relieve people from feeling at a loss when starting a big project with ambitious goals.

What to achieve in one development phase is only made clear right before the phase starts. At that time, the performance of last few phases is evaluated, and the reasonable amount of work is assigned to each member. Thus, the tasks are more possible to be completed, and it will not put too much pressure on the team.

* + - 1. Levels of Accuracy and Units of Measurement

As the course regulates, the project participating measurement is hours: for each student, 30+ hours a week on average should be given to this project.

As a risk control strategy, we give everyone buffer time at least one day a week. It will be discussed in the risk handling section. Our evaluation of a good scheduling is allowing a member to complete his work easily but still have sufficient time (estimated 6 hours per week) to correct mistakes or just do something else for fun.

However, the fact is that the developer always has heavy burden and spends more hours to complete his work. In the meantime, he has to wait for a long time for the design to proceed. The designer always put off his work, and finally the developer has to design himself. Administration documents are not often completed and submitted on time with high quality, and the other guy has to take this responsibility later on. All the delays have given the leader very negative feelings of his managing ability.

This situation continues all the project time, which makes the planning much harder especially when the academic deadline is coming. The only way to get everything back to normal is the other guy’s much harder working. To conclude, the accuracy of our planning is pretty low.

**1.2.4 Responsibility Assignment**

The basic rule of our planning of assigning work is

* Firstly giving each the tasks that meet his skill set; and,
* Secondly scheduling the tasks as parallel as possible to avoiding deadlock where one person cannot proceed waiting for the other’s completion of work.

The original plan is that the experienced member makes plans, monitors the progress and controls the risk, and also does the technical stuff; while the other is responsible for designing pages and databases, and then produces administrative documents.

In fact, as described in the personnel section, one of the team member has past IT working experience, while the other is not a software engineering student. The project coordinator left the team a big trouble when introducing the networking student into the team and gave no permission to refuse. It was quite ridiculous, and it was the only reason for the team to rework the proposal and wasted much time planning and waiting for unhelpful work which was always put off.

Well, life has to go on and problems have to be solved. In order to succeed this project, the other has to take on the responsibility and work more hours to make up for it.

**1.3 Design**

Our goal is to build a website for selling computer parts and accessories.   
Indeed, there are already mature, in-use websites we can refer to when starting our design, but in fact we do not have that many people and amount of time.

Furthermore, ReactionCommerce, as an existing platform which already has its several versions released, does have its own basic design, although we do not like it very much (this design can be found in our development manual or on the official website). But if we change it too much, it is highly possible that we will not complete the task on time and put ourselves into trouble.

The compromise is keeping the original interaction and layout from Reaction (for example, headers, footers, login buttons, cart icon, and so on), but change the styles of pages, like colors, fonts, pictures. In this procedure, “Home”, “Product”, and “About Us” are the only customized pages (keeping the page layout, just change some elements, the description can be found in the development manual).

The importing product feature is the one that our client wants most. Thus, we must find a way to accommodate product lists from different suppliers into current database design, and it is the most important part of our successful project.

**1.3.1 Interaction Design**

As mentioned above, ReactionCommerce has a mature design of almost every functionalities that a shopping website developer wants, such as page layouts, router, dashboard, and many useful libraries like payments and internationalization. This platform is awesome, but we think it is a little strange and it is different from what we are looking for. According to our current requirements, we cut off the following features from ReactionCommerce:

* Multi-shop support;
* Tags;
* Internationalization (translations, dates, time zones, currencies, countries);
* Product grids;
* “Add Product” page;
* Currency exchange rates;
* Payment methods (Braintree, PayPal, and some others).

Add we also add extra pages:

* Home;
* Product;
* About Us;
* Import Product.

**1.3.2 Page Style Design**

We are very painful when doing our design. On one hand, we know that the page design is a very important factor that makes a website successful and we want to do our best to fulfil that. Fantastic pages also make developers feel delightful and improve our development efficiency somehow. But on the other hand, none of us has the experience of designing a good-looking page, or ever uses software like Photoshop to edit pictures.

Some online websites, such like Wix, provide web page templates which are pretty beautiful. We like them, but we do not think they can be incorporated into Reaction framework seamlessly. To make them right for the shop, we must be supported by some experienced guy for pictures, colors, and things like that, which is not within the shop owner’s budget anymore. He does not ask for it in the requirement list, either. So we just give up this idea.

**1.3.3 Database Design**

ReactionCommerce uses MongoDB as its database engine. We only focus on the collection design here, and the technical details are discussed later in the technical chapter of this report.

In database using SQL language, database design means designing tables and relationship between tables. But MongoDB has another set of concepts. Collections in Mongo are the same as the idea of tables in relational databases, and our task is to modify the existing “Product” collection by Reaction for the requirement of importing product list.

The original “Product” collection has more than twenty fields and some of them are not useful for our suppliers’ products anymore (but can be kept and do not affect our design). What is worse is that different suppliers have totally different formats of fields which makes the normalization of tables quite hard if we use relational databases.

Fortunately enough, Mongo is a schema-less database, meaning documents (records in relational database) in one collection do not need to have the same fixed set of fields. This feature makes our lives much easier by enabling us to put different kinds of products into one collection without worrying about inconsistency and too many null values, which conditions is avoided intentionally in relational table design.

Meteor has the “Simple-Schema” package as a schema mechanism. But it works mainly for value validations, and will not change the schema-less essence of Mongo.

We are pretty successful in this process. The product list fields are different to each other too much. We tell the client about the situation, and ask him, sometimes convince him, to simplify the fields by selecting the fields he really care about. In our case, prices and stocks are the most important information to the client, so we just abandon fields like “thumbnail” and “onsale” which are much less important and only are specific to a certain supplier.

**1.3.4 Problems**

The main problem in front of us is whatever the designs are, they never help the developer. The guy who is assigned to design always fails to provide a useful design on time. At last the developer has to design himself, spending more time than scheduled and putting much pressure on him. The mistakes the designer has made are:

* Search online for definitions and solutions of general purposes, rather than doing it based on the ReactionCommerce platform, which makes his designs not applicable;
* Copy definitions only rather than knowing exactly what is going on;
* Misunderstand the instructions, waste time in irrelevant work;
* Keep putting off the work and miss the deadline.

Because he is not majored in this field, so no one in the team blames him for that. But we are still not skilful enough to solve a similar situation. It is what we must find out in the future.

**1.4 Communication**

As for team communication and cooperation, we think we have done a good job. The team has organized meetings with client and advisor. And we have kept our communication channel all green and can contact each other whenever we have issues.

**1.4.1 Meetings**

* **Frequency of meetings.** Team meetings and advisor meetings are held every week, client meeting is held at the end of every phase. The well-scheduled meetings are helpful to synchronize information and progress, thus making ourselves professional and trustworthy to the client.
* **Meeting lengths.** Each meetings is kept one hour long maximum. Usually the meetings are finished sooner than that.
* **Meeting records.** All the meetings cut to the chase and notes of agreements are taken.

We all very happy about the meetings, and we all agreed that our meetings to be effective and efficient.

**1.4.2 Information Sharing**

We have a couple of chances to share information, because we are supposed to meet each other several times each week, such as meeting time and team working 2pm – 4pm every day. We intentionally make it so to reinforce team interpersonal relationship and communication.

But we got problems. The main reason still was our mismatch of available working time. Once a time, we had reached an agreement during the meeting time, say, modification of some part of one document which should done by Vineet, and we were supposed to check that next day. But he did not complete it before next meeting for some reason due to his work. And he just did not tell the team and even worse did not go to the meeting we appointed. It made the other teammate confused and the work was delayed, affecting the progress of the project.

Sometimes he got errors when running the project, but he did not say, and no one knew what was going on. But he again did not tell anybody until he missed the deadline and it was too late to help him. We talked about it, he admitted his mistakes but he never changed.

**1.4.3 Instructions**

The instructions were not delivered well for this project. Instructions were often forgotten or misunderstood, which led to unhelpful work and more time to explain and rework.

Members were told to take notes about what to be done. At first this helped much, but later this habit was not kept anymore, tasks forgotten, documents forgotten or made in wrong formats, passwords lost. Such kinds of small but annoying mistakes happened all the time.

**1.5 Monitoring and Controlling**

**1.5.1 Progress Monitoring**

Officially the progress is examined every week during the team meetings when we check each other’s work and fill in the progress report. We did it very well not only because we performed this on time but also we gained much from it by finding what was going well and badly and the reasons why. It can be seen as a serious way to report performance to the teammates and advisers.

At the moment we produced the progress reports, it was really tricky to think up why we were doing well or not. It was easier to tell why things were not good enough, since different issues resulted from different reasons, and the reasons were relatively specific, like “not enough background knowledge”, “got errors when building the project and spent hours to search for a solution” or “hurt by three local drunk men and stayed in hospital for a week”. But when it came to why good was not that direct. Being self-disciplined could be a frequently-used reason, and it seemed that nothing else was applicable.

**1.5.2 Quality Control**

Reaction is maintained and fully tested by an open source group. Adopting such platform, we can ensure the product high quality that is enough for an online commerce environment.

1.5.2.1 Standards

Our quality control process is mainly for customized pages. Usually, there are strictly specified standards describing how to define the production quality, for example, passing rates of test cases.

Reaction (as a Meteor project) has a built-in automatic testing framework, which is already employed into the platform. But we do not have time to study and implement testing code for our features, and it can be planned in the future.

We choose manual testing, and the test cases are split into modules (pages). All the test cases must be passed and then we are able to say that our product is good enough to release.

1.5.2.2 Test Planning

* Unit tests occur in the 4th and 5th phases. At the end of phase 5, all the page development should be completed and test report is dispersed;
* All the test cases should be passed. If any defect is found out, plan should be made for phase 6.

1.5.2.3 Test Result

All test cases are passed. And the tester has given some advice for future improvement.

**1.5.3 Change Control**

As our methodology is quite flexible and change tolerant, it is quite possible to arrange changes during the project life cycle. What is more, our schedule is designed into phases, and changes are acceptable at the beginning of each phase, since incremental and adaptive models are particular for that purpose. Also, our requirements are put into product versions. In each version, one requirement is developed based on the completion of requirements from previous versions. That means, there is no overall design existing and plans are made for individual versions, so there is actually no scope change.

The changes we have are mainly related to the processes, such as changes of document formats, changes of meeting timetables. They are all internal changes and do not need change control requests.

**1.5.4 Audit**

The first audit was held on 5 April. Robert helped us found some problems of our processes and documents, and the findings were:

* Did not have deep understandings of some details of meetings agendas and minutes, for example, “action items” were misused;
* Item resolutions, reasons or agreements were not specific and detailed enough;
* Did not keep records of informal client communication;
* Did not record administration time;
* Individual logs did not include personal-interpersonal side.

The second audit was on 20 April. We solved all the problems found in the first audit and got a good mark.

**1.6 Things That Happened Unexpectedly**

**1.6.1 A Teammate Joined**

The team only had one person in the first phase. Right before the second phase came, Vineet was introduced into the team. This affected the project in two ways:

* At that moment, the project proposal had been made, and the plan was confirmed. It meant that the time we started was delayed and we had to spend more hours to modify the schedule and figure out what and how he could contribute to the team.
* He did not have the appropriate skill set, which meant that it was really hard to assign him any work to do. What was worse, the expectation of project outcome from the client was apparently much higher with more men in the team.

The solution was, sadly, to put up with it. But to see it positively, they did Xiaochen a favour by putting him into a team, and Xiaochen did Vineet and Robert a favour by accepting him (although no chance to refuse).

**1.6.2 Change of Framework**

Xiaochen originally chose Angular + Strongloop to start with, and he already wrote a demo and showed to the advisor. However, Robert did not agree to this plan because Xiaochen wrote in the proposal “build from scratch”. They apparently had different understandings of “from scratch”. The team was strongly asked to try some mature solution instead.

This was at first not acceptable to Xiaochen, because he had been researching that combinations for two months. But later he indeed found a framework called Meteor and an e-commerce solution called ReactionCommerce. It was totally JavaScript-based framework, and the previous JS study time was not a kind of waste anymore. It affected the project in these ways:

* Rework of research cost us more time, but instead, we got a workable website more quickly. In this aspect, the change did not affected the development time too much.
* Helped the team find a mature JavaScript-based framework, which was very useful for future projects.
* The sudden change of solution made the client not very happy.
* The original demo was not useful anymore and the research was wasted.

**1.6.3 Reaction Version Migration**

After our research of Reaction and the development was about to start, Reaction version 0.13 was released. This version affected out development in two ways:

* It used another routing package, which was totally different from the one we had spent hours to study. We have to learn this new package from start, and the notes we had taken were not helpful anymore.
* This version was not stable and many of the users got error when running on the browsers. And none of the Reaction team had ever found a solution.

The solutions were, first, gave another three days to study the new routing package; and second, also planned hours to find the reason why browsers showed errors all the time. Actually we found the reason and fixed it. We were very happy about the result.

# New Technologies

**2.1 Meteor**

A fantastic JavaScript based framework for web and mobile applications. It provides developers a bunch of tools and libraries to facilitate the development. When one is researching which framework to use, consider the following points:

* One language for all applications. Especially since ECMAScript 2015 was released, JavaScript standard has been accepted by many frameworks, such as Meteor and Nodejs. And Meteor includes a package that can translate ECMAScript 2015 syntax into the normal JavaScript syntax which reinforces browser compatibility of the applications.
* Built-in MongoDB support. Many packages are employed to make the reactive design possible and easy to implement. “Simple-Schema” packages allows developers to complete the value validations for Mongo.
* The ability of building the applications into different platforms. Meteor also enables developers to build projects into Android or iOS applications, which is a very powerful tool for cross-platform deployment.
* The possibilities of introducing Angular and React libraries. Apart from the built-in Blaze rendering mechanism, Meteor also allows Angular and React rendering.
* Meteor supports CSS interpreters. Formats like .less are supported and the files are built automatically when the projects are being built.

**2.2 MongoDB**

If there is no history database to be compatible for, MongoDB can be an option when considering what database engine to use. It is a quite flexible engine and the collection design has a different strategy from that of relational database engines like MySQL.

The schema-less feature demonstrates its power in our collection design where we combine variant fields of products from different hardware suppliers in an easy way. It will be much more complicated if we implement the design using a relational database engine.

**2.3 Docker**

There are several ways to deploy a Meteor application. For example, we can deploy the application to Galaxy which is run by the Meteor officials; and, we can also deploy onto Amazon Web Services. And the best way to deploy onto a local server is to use Docker, which is suggested by the Reaction team.

Docker is another big topic. It is a good way to deploy and scale servers, and in the meantime it can keep server instances, load balancing (Nginx) and database engines separate to each other in the same computer hardware.

A Docker container can run anything. An operating system, a program, or several lines of commands, in a separate space. With the help of Nginx, we can run multiple Reaction instances in one computer, thus increasing the traffic loads of our server to accommodate more user requests in the future.

Ibuyit has not used Docker for this moment, because we have not solved the version problem of node-aes-gcm module. But if we have future plans of deploying this to production server, it is a must-have process.

# General Recommendations for Future Group

* For any project, a mature framework is always a good start, especially when the developers are new to the programming languages, frameworks, or platforms.
* However, the framework is usually not quite customized for the project, and they cannot compete with the currently in use famous website, like Amazon, Taobao and some others.
* Think carefully whether to use a certain technique. Website includes many aspects of techniques, from back-end framework to server, and the project will possibly fail if any of them does not work properly.