## Assignment 3 Project overview WIP – TP 31/10/21

Below tied up for Thursday this week – UI prototyping an ongoing project until Next Sunday, should be wrapped by then (hopefully).

### Plans and Progress

Here you should give as much detail as you can about what your project will do, and how you will do it. This should also include how far you have got with developing any features or outcomes from your project. Tell us about the “story" of your project – how it began, how it has progressed, and what stage of the plan you are up to. Include any dead-ends you may have followed, decisions made, and changes that have been made to the project plan. This will need to include a significant amount of detail, so that it is easily seen what precisely you have done and are planning to do. If it helps, imagine the information that would be required if you were to hand this project over at the end of the semester to a new team to complete the job. What would you want to know, if you were one of the people taking over? There is no set length for this section, but it is hard to believe that less than two pages could be adequate. Three or four pages is far more likely.

The Problem

The Solution

The Method

* High staff turnover rates, seasonality employment, low margins/high staff cost
* “Small and medium businesses with higher levels of digital engagement are significantly more likely to be growing revenue, creating jobs, exporting and innovating new products or services. Despite these benefits, many businesses are still a long way off adopting digital technology”
  + Department of Industry, 2018. Australia’s Tech future. [online]p.17. Available at:

<https://www.industry.gov.au/sites/default/files/2018-12/australias-tech-future.pdf>

[accessed 7 November 2021]

The highest cost for a business is staff. This is especially true in the hospitality and retail sectors, where seasonal and low-skilled employment is often used to bridge staffing gaps over high turnover periods, such as Christmas. These industries are challenged by low profit margins against comparatively high staffing costs, making it difficult for businesses to generate consistent and strong revenue to enable the growth of the business.

According to the Department of Industry’s report from 2018, “Small and medium businesses with higher levels of digital engagement are significantly more likely to be growing revenue, creating jobs, exporting and innovating new products or services.” Despite this, “many businesses are still a long way off adopting digital technology.” We attribute this to lack of intuitive, affordable, and accessible solutions for small to medium businesses; it is for these reasons the idea of stockIT was created to bridge the gap between the demands of the industry with the skills of the workforce.

StockIT is an integrated inventory management and business platform. It aims to empower business owners to make smarter, evidence based decisions through access to real-time data on the purchasing, tracking and fulfillment of inventory. It is through this process that stockIT’s three main goals can be realised, which are to:

* optimise wastage
* increase purchasing power
* improve supply chain efficacy.

In order to appreciate the full value add of stockIT for a business, it is important to understand the basics of and the underlying mechanics of inventory management. Purchase, store, supply, sell and analyse

INSERT IMAGE FOR INVENTORY LIFECYCLE

Digitalising inventory management by using stockIT a business can optimise their resources, increase purchasing power and improve supply chain efficacy.

The first iteration of stockIT was an app idea named “Stock-Take-It”. Initially the app was designed to fill a need in a group members business where productivity and efficiency were being lost during the inventory management process, The app focused on a simple easy to use interface and on increasing the user’s efficiency when it came to counting and collating inventory data.

The stockIT we see now grew as an extension of the original seed of “Stock-Take-IT” after brainstorming concepts and ideas in a group setting. The change came about through open and constructive conversations about each of our experiences with inventory management, business development and functionalities we would like to see in software suites aimed at businesses.

Assignment two provided our group with the opportunity to explore the concepts and core functionalities of stockIT in an unrestrained, open-ended manner. We were able to freely explore functionalities without a need to narrow down and focus only on what was achievable given the semesters timeline. While we did keep in mind the requirements and restraints of Assignment three when formally writing our report and detailing functionalities, we did not want to restrict our flow of ideas simply because of time constraints.

A key motivating factor in the creation of stockIT was for us, the development team, to be able to create a product or software suite to bridge what we perceived to be as a gap in the current market. While we did and still do acknowledge that there are products on the market today that cater to similar functionalities and business requirements (please see landscape section of this report for more information), our belief is stockIT provides far more beneficial functionalities, better scalability and is far better suited to a retail or hospitality customers needs than the competitors.

By not constraining our ideas for assignment 2 we created two problems. The first problem was “feature creep”, whereby as the idea grew from Stock-Take-IT to stockIT more and more features and functionalities were added to the software. This is a problem because it creates a situation where we can possibly lose sight of the fundamental and unique features of the software as we try to implement too many features. As more features are added, it becomes increasingly difficult to distill the software and marketable product down to its core features.

The second problem was that by fleshing out the concept of stockIT to such a degree, we had a greater level of difficulty in pulling the focus back to deliverables for our third assignment. As the software of stockIT grew and developed in our minds, so did the amount skill and experience required to offer up functioning deliverables.

However, with problems come solutions. Our solution to the issue of feature creep was to limit the scope of deliverables for assignment three. Limiting the scope of the deliverables meant that we could better plan out the future features and functionalities of stockIT and spend more time laying the foundations for the projects success outside of an educational environment. It also, I believe, bought the group together in a way that shifted our focus to a more goal orientated way. No longer were we looking to come up with new and exciting additions, now we were looking to build upon the ideas we had and work out ways to implement them going forward.

Because of the vast undertaking that is the development of a fully functioning version of stockIT, our deliverables for assignment three may feel that they fall a bit short of where we as a group would like them to be. We are realistic about our skill level and the actual skill requirements to develop a fully functioning software suite, as such, our key deliverable is our wireframing and prototype user interface for both the WebApp and Mobile app version of stockIT.

Originally we planned to create our mobile app prototype by using the MIT app creator. After the development group spent some time testing the software and its features we made the decision that given our 6 week timeframe for an initial deliverable, the time investment required to take full advantage of the MIT app creator would not yield the results we wanted. The MIT app creator would be better suited towards a small mobile game or an app with less required functionality. The developer experience will using the creator was found to be somewhat clunky and cumbersome to navigate while a lack of collaborative features meant that working on the development of the app with one or more people would become a logistical challenge.

At this point it was decided that we could not pursue the development of our mobile App using the MIT App creator and further research would be required. After brainstorming the key features that we wanted to present in our first deliverable (and the dead end that was the MIT app creator), the development group came back to refocus on one of the first core principles of stockIT, a clean and easy to use user interface. It was after this refocusing that our attention shifted toward using Figma. Figma would allow us to create our ideal user interface, present an animated prototype and work on the project in a collaborative manner. All of these were important factors for the development team in our choice of wireframing software as it was fundamental that the capabilities of the software aligned with our goals and motivations as a development team.

A further functionality that the development group wanted to explore to work in tandem with our first wireframing deliverable was to demonstrate a stockIT functionality running using a programming language. As the programming language skills of the development group are quite limited and Python is the language that most group members have some experience with, it was put forward as the option for us to explore. The plan was to make a program whereby we could create an inventory list, input the amounts of each item in the inventory list and then have it be displayed in an informative way for the user. Whilst Python is perfectly capable of creating modifiable lists, another defining feature of stockIT is the ease of use and locations/storage zones and it was decided that while Python could be used to create a list feature, it would not function in the same easy to use way that the development group would have liked in an end feature of stockIT. Therefore to avoid delivering what we felt was a sub-par and non-reflective artifact simply for the sake of it the development group decided to re-invest the resources into alternate areas of project development.

Our goal for the very first development plan of stockIT was to give potential investors/marker an overarching view of the capabilities and functionalities of stockIT. For this reason, our first report was a very weighty deep dive into what the development group believe to be the fundamental selling points and features of stockIT, we consider this to be our wish list, and it should be viewed very much as a “What could be possible with a large budget, the required skills and resources”. For our second iteration of the development plan we are taking a much more strategic, refined and realistic approach to the future development of stockIT.

As mentioned, the key deliverable now is our wireframing and prototyping. The development group believes that this best shows the capabilities and trajectory of the software, while remaining in our current skillsets. Alongside this, a comprehensive list of future features (non-deliverables) and their reasons for exclusion at this point in the development cycle can be seen below and in the project timeline included in this report;

|  |  |
| --- | --- |
| **stockIT Future Feature** | **Examination** |
| Software Integration   * MYOB, XERO, Vend, Square etc | Integrating software can be an arduous process. Not only do we have to navigate the software being able to function on a code by code basis and facilitate the communication between the two separately developed entities but we would need to navigate the legal and functional business logistics of both parties. |
| SILo Feature Functionality   * Collate inventory data to single repository * Collaborative users * Ties into Cloud and local storage | An overview of SILo’s and their functionalities can be demonstrated in the Figma wireframing and prototyping however SILo’s and their coding require a greater level of skill and experience than we as a development group possess. We can illustrate how they will operate and function in hypothetical sense, but not in a demonstratable (backed by coded functionalities) fashion. |
| Supply chain integration   * Supplier lead times * Raw good to Finalised product | This process relies on a coded and realised version of stockIT and the underlying function here is to have two separate instances of stockIT be able to relay and communicate information between each other. |
| Storage Database infrastructure   * Cloud storage * Local storage | A vast understanding and working knowledge of cloud and local machine infrastructure, SQL, DDL and DML languages are required to implement these features. Having inventory information be storable and transmittable both in a local fashion and through the cloud is key to the operation of stockIT. |
| Reporting Functionalities   * Sales data * Predictive and historic data * Purchasing Information * Accounting information | Much like SILo’s, the presentation of this information can be demonstrated through Figma wireframing however a functioning prototype of these features cannot be created without a working knowledge of both SQL and a programming language like C++. |
| Direct Purchasing Integration | This feature executes in a similar fashion to supply chain integration – a working build is required to execute. |
| AI Integration   * Predictive behaviours * Trigger alerts for low stock * Purchasing and supplier triggers | The end game feature of stockIT and what the development team all agrees will set the software apart from its competitors. A fully functioning build of stockIT is required to implement this feature with extensive in-field testing to be carried out for data gathering and analysis to be done before it can be even implemented in a Beta phase. This is likely a feature that will run in the background of a “finalised” stockIT build before we can advertise it as fully formed feature of stockIT. |
| Unit Profiles   * Stocking keeping formulae * Recipe information * Accounting information | Unit Profiles are demonstratable in a purely aesthetic sense through Figma wireframing. While we can provide real data to use, the core functionality requires a high level in databasing language and programming language skills to implement Unit Profiles as a functioning feature. |
| Stock Taking functionality   * Export data to CSV * Multiple Users | While the development group did experiment with and brainstorm the use of Python as a functioning prototype for stock taking and inventory counting purposes ultimately we deemed the language and our execution with it would not meet the standards and requirements we had for it. As such this is a work-in-progress feature and is likely one of the first items we will implement in a working build to facilitate the growth of the app and the database building features of stockIT. |
| Function Mobile App, Web Application and standalone Program for Windows OS | Figma wireframing and prototyping can give us a very good indication of the end product look and feel however a functioning prototype for the Mobile app, Web App and standalone windows OS program are not feasible until our skills develop or outside help is acquired. The Mobile app is first in the development cycle as it will provide a means of advertising the product with very limited functionalities and features and is a good way to step into the market. |

The above table is an accurate list of the yet to be implemented features for stockIT. While the features do present what at first may look like an insurmountable challenge, it is one that we as a development group feel is achievable – although not in the initial 6 week timeline we have been allocated. Therefore, we are using this project both as a learning exercise in entrepreneurship but also as a way to view real world roles and jobs in the IT world through a more focussed project lens. Long term, provided we have the resources, time, skills and financial backing stockIT is something that we believe is a very realistic and achievable project and business idea. Short term, we have had to reign in our expectations about what is achievable and actionable. It is all very well to plan out a large scale software application like stockIT on paper and list out all these great, intuitive and exciting features however when you take the time to plan out how each and every one of these features and functionalities can come to fruition the task can quickly become very overwhelming.

Throughout this project the development group has been simultaneously excited and terrified at the prospect of creating stockIT. The real world potential for execution is there, but seeing the amount of resources required to create a project like this has meant we have all been quite humbled by our actual skills in the IT world. We have learnt that we can achieve a great many things when we work together, creating a concept and fully fleshing out the intricate details can be a very difficult thing to do when working as a group – especially when all the work is carried out virtually and amongst a group of total strangers. Over the course of the experience we have learnt to rely on each others individual skillsets and be responsible for our own allocated tasks all the while being realistic about what we can actually achieve given the nature and timeframe of the project.

Need a conclusion here.

Second iteration

* + How the idea grew/changed/developed from original concept when brainstorming with the group.
  + Mind Mapping
    - Splatter gun shot of ideas and concepts
  + Group involvement changed the idea from stock taking app to what stockIT is now
  + Second assignment was very much a “What could we accomplish if we saw this idea through to the end of development”
  + Development dead ends
  + How overarching and large the changes to original concept were
* Third iteration
  + Deliverables
    - Wireframing and UI prototyping
    - Back end offerings
  + What we can actually achieve vs what we planned to
  + Presentation
    - How does it fit in
  + Is the overall project achievable?
    - Is it workable?
    - Can we make this?
* Deep dive into the possibilities of the project but the limitations due to skill-sets, time and resources.
* Lessons learned between all three stages

## Timeframe

Another difficult aspect of project planning is knowing how much time to allow. You will have something like 36 hours per person for this assignment. In order to develop a plan for further work beyond the end of this course, let us assume that you will have an extra 10 hours per week per person for 10 weeks in addition to this time in order to develop your project. This means that you will have six weeks (Weeks 7 to 12) of the semester to work on your assignment, with a further 10 weeks after that. This means that your plan will be for a total of 15 weeks, with the first 6 being on this assignment.

You will clearly not have the extra 10 weeks to work on the project; this is intended to give you a feeling for how much you would be able to achieve in that time. This means that the first 6 weeks of your timeline will end up being your actually progress on this project, with the remaining 10 weeks being your plan for the next stages.

This should be presented in the form of a table, with one row for each week, specifying as best you can the work for each person for each week. This means that the first six rows of the table will describe your progress so far, and the remaining 10 your best guess at how the remaining time would work.

This will no doubt change as you work on your assignment, as it will give you a more precise idea about how long it will take to get things done. This is not an unchangeable contract for exactly how things will work; that is unrealistic for just about any project. The idea is to get you thinking about how exactly your time should be allocated to the various tasks involved. It is a good idea to have a milestone (i.e. a specific outcome) for each week of the project. This may include getting familiar with tools, or reading up on a particular technique or technology. You should also include time for writing up the final report and any other documentation. Writing reports always takes longer than you think, especially as you should expect to re-write any piece of writing that you do at least three or four times.

* Visual time scale – mind mapping it
  + Visual time line with key points and deliverables
* Excel spread sheet the week by week deliverables
  + Similar to Tetsu’s sheet from second assignment
* What we **can** achieve in our given time frame
  + 6 weeks vs 10 weeks
* What we **can’t** achieve in our given timeframe
  + 6 weeks vs 10 weeks
* Report writing timeline
* What we could achieve given resources, experience & more time.

What are we trying to achieve in the 6 week timeline?

* Working UI for Web app and Mobile app
  + Unit Profiles
    - Recipe information
    - Allocate suppliers for each product
    - Inc formula for tracking
  + SILo’s
  + Accounting section
  + Supplier detail section
    - Create orders from Supplier section
  + Purchasing information
  + Input actual stock datas from RSB using the stocktake
* Video presentation of the App for investors/Public/Crowdfunding
* Roadmap for the design and implementation of the product
* What we can achieve as is VS what we could with technical skills
  + 16 week timeline is as if we hired a team of Devs
* Business model + Subscription Model
  + Generate prices
* Competitors
* There is a strong difference of what we can achieve as a group doing an assignment VS what we can achieve as a development team creating a product.

What are we trying to achieve by the end of the 16 week timeline

* Working back end code
  + Products can be inputted
* UI Figma Web App presentable on HTML website
* AI barebones/functionalities listed out
* Chatterbot integration
* Fundraising and investment pitch

## UI Development and Prototyping

* Time-line for development
  + Where does this fit into the larger timeline
* What’s realistic
* What should we show, what shouldn’t we
* Different ideas – best of the best UI from the opinions of each group member
  + **Please give me your inputs – I’m stuck looking at the same websites and need some fresh eyes and fresh ideas of good elements or places to look.**
* How much back end do we need – how does the stock data from RSB fit into the UI?

## Risks 5

What risks can you identify for your project? There will always be some generic risks (such as computers breaking down the night before a deadline, health and family issues, and institutional changes). Do not include generic risks such as these. The idea is to be as specific as you can to your project. For example, if your topic is to develop a game, there may be a risk that the software you choose to work with may be very difficult to learn, poorly documented, or not turn out to have the features that it claims it has. These properties are often only discovered once you have started working with the software, and so unless you have had lots of experience with the particular tool, there is always a risk that it may not work as well as you believe it should, no matter how much prior research you do. Similar comments apply to hardware.

|  |  |
| --- | --- |
| risks | |
| Assignment | Project |
| Market - landscape | Financial |
| Temporal – deadline | Market |
| Software | Temporal – Investor contracts |
| Skills | Supply Chains – Covid etc |
| Hardware | Staffing |
|  | AI |
|  |  |

What is the risk, what is our response, Can we mitigate Yes/No, How have we

Shared risks

* + Market
    - Will the market change, grow, etc. What is the risk, what is our response?
    - Small/Med Business data from above
    - Business Wire inv data from assignment
  + Temporal
    - Time to complete project
    - Time to complete assignment
    - Time is a factor in any situation, especially noticeable with deadlines and project dev lifecycles
    - First to market
    - Similar products at the same time, want to be first with the new niche’s

Non-Shared risks