**PROJECT INCEPTION**

Vibhas Kamal

Isara Arunanondchai

Jack Beard

Manvendra Singh

Quang Ly

**PROJECT PLAN**

**1.1 TEAM ORGANIZATION:**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEAM MEMBERS** | **CONTACT NUMBER** | **EMAIL ID(@student.monash.edu)** | **ROLES** |
| Vibhas Kamal | 0451 621 080 | vkam0002 | Product Owner, Developer |
| Mike(Quang Nghiep Ly) | 0420 819 099 | qlyy0001 | Developer |
| Jack Beard | 0400 926 760 | jbea0002 | Developer |
| Isara(Tito) Arunanondchai | 0481 944 555 | iaru1 | Developer |
| Manvendra Singh | 0450 798 864 | msin0017 | Developer |

**1.1.1 Roles and Responsibilities:**

* Product owner:
  + Availability: 3 hours on Monday/Tuesday (beginning of each sprint)
  + Tasks: finalizes each sprint's product backlog before the Sprint meeting on Monday/Tuesday
  + The main responsibility of the product owner is to represent the interests of the client and act as the middleman between the team and the client
  + If any there is any confusion regarding the requirements and the team members are not able to resolve it, the product owner would be contacted and would be responsible to resolve the confusion.
* Team:
  + Availability for contact from 6pm - 11pm.
  + Submit code regularly (ie. should commit parts of tasks or tasks instead of finishing all allocated tasks then submit everything in one go)
  + Submitted work needs to meet the definition of done
  + Need to support one another in the case of technical uncertainty

**1.1.2 Team’s Process Model:**

* Weekly meeting held twice (Wednesday, Monday/Thursday):
  + Purpose:

These meetings will be the Scrum meetings. It will include discussion of to-do work, assigning tasks, and possible re-allocation of tasks to team members (in case certain tasks turn out to be much easier or harder than expected).

* The sprint duration will 2 weeks with a sprint retrospective at the end of the sprint (which will be the last day of the sprint)
* Checking progress of assigned task checked in the last meeting.
  + Upon completion of every task assigned in one sprint, update sprint backlog and make corresponding changes on Trello.
* Sprint retrospective after every sprint.
* Formal consultation with client each Friday and contact through email (if needed) during the week.
* If the team members are facing any problems regarding the requirements, the team would be consulted. If the issue is still not resolved, the Product Owner would be contacted immediately. In case the Product Owner does not have the relevant information, the client will be contacted and the work on that requirement will be halted, and the developer working on that task would be asked to move on to the next task.
* Most of the communication within the team would be done via Facebook Messenger. If there is anything that needs to be clarified or conveyed immediately, the team members would use the contact numbers of the team members to contact the intended person.

**1.1.3 How the team’s process model differs from Scrum:**

* Group members do not have a daily meeting (Daily Scrum). Meetings are held twice a week (at least).

NOTE: In the team meeting, it might not be possible for all the team members to be present. So, it is possible that a ‘team meeting’ might have just 2 team members present.

* In addition to process review (sprint retrospectives), we will also run a technical review on the code produced within each sprint, which would be the sprint review. This will help us to plan our next sprint.
* Rotation of Scrum Master role through each Sprint:

In the conventional Scrum process model, the person undertaking the responsibility of the Scrum master does not change. However, in order to allow all the team members to learn the Scrum process model properly, two team members would be asked to be the Scrum Master (each for one sprint (minimum)).

**1.2 TIME AND TASK TRACKING:**

**1.2.1 Allocation of tasks to team members:**

* During the team meetings (scrum meetings), everyone will give their preferences regarding what task they want to do. After a discussion on the task allocation, each team member will be provided with tasks on the basis of the team member's abilities, interests, knowledge in that area.
* If a task exists that no team member wants to do, the team would meet together and try to figure it out together during one of the team meetings or will allocate a separate time which suits every team member’s availability.
* These tasks will then be formally allocated on Trello to the team members.
* However, it may be possible that the allocation would not happen at all due to the functional history of the team. The team history, in terms of task completion and work on a task, has indicated that the team functions best when all the team members are working on the same task together and are sharing the progress that each team member is making with other team members.
* There is a very high probability that the team will work according to the (immediately) above mentioned point. The reason being that due to the complicated nature of the task that the client has asked for, the team members would not be able to work on separate tasks to achieve the final goal because all the tasks are somewhat related to each other.
* However, team would make an effort to split the tasks and allocate it to different members because it would safe to assume that everyone working on the same task at the same would lead to confusion and uncertainty and it would ultimately prove to be unproductive.

**1.2.2 Tracking of time spent on project tasks:**

* Google Spreadsheet will be used to keep track of the hours each member spent on doing the tasks related to this project (based on the assumption that no one would falsify information). This would be shared with the marker at the end of the project or the marker would be given access to the spreadsheet if required.
* If in case a team member is not able to complete the task in the allocated time, an extension would be given for that task based on what the other team members think about how much more time that task should be given.
* Our team is using an effort-estimation model to estimate how long a task might take to complete. These efforts are being measured in the form of T-Shirt sizes. The table below indicates the ranges of values (effort) for each T-Shirt size:

|  |  |
| --- | --- |
| **T-Shirt Size** | **Range of Value (Effort)** |
| XS | 0-2 |
| S | 2-4 |
| M | 4-6 |
| L | 6-8 |
| XL | 8-10 |

**1.2.3 .Tracking of progress on your project:**

* We will use a web application, Trello, which will allow us to track personal and team progress on individual and group tasks.
  + The marker will be given access to Trello if required.
* In addition to the formal use of Trello, we will employ the casual strategy of communication through social media for instant delivery and retrieval of milestones related to the progress of our project.
* The sprint backlog is another artifact that can be used to track the tasks that are supposed to be completed for the sprint.

**1.2.4 Storing and managing backlogs**

* Vibhas Kamal (Product Owner) - Product Backlog/Sprint Backlog
* Remaining members of the team - Sprint Backlog
* The product and sprint backlogs will be pushed on Git and will be uploaded to Google Drive. This is done in the case we need previous Sprint’s data for progress monitoring.
* Sprint backlogs will be updated at least once a week (if needed) so team members know the current state of the sprint.
* The main reasoning behind the need for varying editions of the sprint backlog lies with the focus on agile development, whereby if the requirements of the project are susceptible to change, then the overall list of requirements can have tasks that the team has previously agreed on to be removed, or added with a newly refined definition.
* With the lack of flexibility in sprint backlogs, it proves incredibly important that these changes are made apparent before the sprint has begun.
* HOWEVER, if a situation is to present itself such that the requirements of the end product have changed, the exception of updating the Product Backlog and or the Sprint Backlog would be made during the sprint.

**1.3 DEFINITION OF DONE/COMPLETE (CODE/DOCUMENTATION):**

* Produced code for presumed functionalities
* Assumptions of User Story met
* Project builds without errors
* [Unit tests](https://apiumhub.com/tech-blog-barcelona/top-benefits-of-unit-testing/) written and passing
* Project deployed on the test environment identical to production platform
* Tests on devices/browsers listed in the project assumptions passed
* Feature ok-ed by UX designer
* QA performed & issues resolved
* Feature is tested against acceptance criteria
* Feature ok-ed by Product Owner
* Refactoring completed
* Any configuration or build changes documented
* Documentation updated
* Peer Code Review performed

1. Software functional specifications and user stories are met, and accepted by the Product owner, whom has ensured that what being delivered is in line with what the client wants.
2. All components within the environment serves a purpose, and is meaningfully integrated with other parts.
3. With regards to the work being done by the team members, all tasks are able to be tracked and traced in order to be discussed in reviews.
4. Code is complete, in the sense that it is: refactored, commented, understandable, meaningful, and consistent. Unit testing must be completed for each component, and integration testing for the parts it is related to.
5. Functional testing is completed to ensure that the deliverables satisfy all the requirements set out by the client. Performance and User acceptance testing also completed afterwards to allow for smooth deployment of the system.

**1.3.5 For other documents of any kind**

* Written the document
* Correctly formatted
* Spell checked
* Submitted for review by other team members (on an individual level) OR review of the document completed by all the team members (on a team level)

**1.4 PROJECT VISION:**

To deliver a web-application that would allow the user to check the contributions of the members of a Google Drive folder/documents in the form of graphical representations.

**ANALYSIS OF ALTERNATIVES (AoA)**

**1. PROGRAMMING LANGUAGE:  
  
Terms of reference:**We are choosing between:  
A. Python  
B. JavaScript  
C. Java  
  
The criteria on which we would be deciding which language to use are:  
A. Experience in the language relevant to the project platform (web based)   
B. How applicable the language is to platform the project is built on  
C. Ease of programming

D. Ease of learning  
 **A. Python:**Our team members have all worked with Python before, and are moderately experienced with the programming language as a whole. However, after going through Google APIs, it was found that the team was comfortable with the instructions on using the API given in Python. Therefore, the team has decided to use Python as one of the programming languages to be used to create the app.

**B. JavaScript:**Most of the team members (4 out of the 5) have used JS to build web-based applications before. Therefore, JS conforms to the knowledge and experience aspects of the criteria.  
The team is planning on making a web app, so the JavaScript is the most relevant language to use out of the possible options.  
The team members have agreed to spend some extra time to brush up their JS skills and have agreed to learn new concepts in the language if the project requires the team to do so.  
 **C. Java:**Some members of the team have lightly touched on Java programming. The experience that they have, however, has nothing related to web based applications. Whilst Java also possesses a web-plugin that allows its applications to be used in a web browser, it is less familiar to the majority of our members and hence, our preference lies with Javascript.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Experience in language relative to platform**  **(Criteria A)** | **Applicable to the project platform?**  **(Criteria B)** | **Ease of programming**  **(Criteria C)** | **Ease of learning**  **(Criteria D)** |
| **Python** | Low | Yes | High | Medium |
| **JavaScript** | Medium | Yes | High | Medium |
| **Java** | Low | Yes | Low | Medium |

**Recommended Programming Language:** JavaScript, Python

**2. OPERATING SYSTEM:  
  
Terms of reference:  
   
We are considering:**A. Windows OS  
B. Mac OS  
C. Linux  
D. Chrome OS  
 **The criteria on which we would be deciding which OS to use are:**A. The relevance of the application with the OS.  
B. Familiarity with the OS in terms of general use  
C. Applicability of the OS to the software being created

D. Availability of the OSThe team has decided to make a web application. This means that the OS of the system on which the app is being programmed or the OS on which the application is going to be run will be the OS of choice. Intuitively, Windows OS is the priority for the application to be built upon, because all the developers are using a Windows OS system, but this is of little importance, considering that each OS listed should be able to handle the web application.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operating System** | **Relevant with OS (Criteria A)** | **Familiar with OS**  **(Criteria B)** | **Applicable to OS (Criteria C)** | **Availability**  **of the OS**  **(Criteria D)** |
| Windows OS | Yes | Yes | Yes | Yes |
| Mac OS | Yes | No | Yes | Yes |
| Linux | Yes | No | Yes | No |
| Chrome OS | Yes | No | Yes | No |

**Recommended Operating System:** Windows OS

**3. PLATFORM:   
  
Terms of reference:  
We are choosing between**A. Web application  
B. Desktop application  
C. Mobile application **The criteria on which we would be deciding which platform to use are:**A. Resource usage intensity  
B. Level of resemblance between the program requirements and the platform type  
C. Level of familiarity with the platform  
 **A. Web application:**Considering that our group has decided to work on the gitinspector app, it would be safe to assume that the resource usage intensity would not be too much and as a result, it can be created on a web-based platform.  
Since the gitinspector that the group has decided to create works on Google Drive, the relation between the program requirements and the platform type becomes easy to understand.  
Since all the team members have used web apps extensively, everyone is comfortable with creating a web app.

Additionally, Google Drive is mostly used as a Web App. So it would be safe to assume that the GitInspector for Google Drive would also be used as a web app.  
 **B. Desktop application:**Desktop applications are undoubtedly the most familiar type of application to every member of our team - their use, in the context of requirements, however, are arguably not as applicable as web applications in the case of GitInspector.   
 **C. Mobile application:**In terms of development familiarity, collectively we are not very experienced when it comes to developing a mobile application. The program requirements would be achievable, but it would certainly be more difficult to do so and whether the team would be able to meet the deadline requirements would be questionable. So in this case, a mobile application would be something that we could extend to rather than something to start on.

**Recommended Platform:** Web Application

|  |  |  |  |
| --- | --- | --- | --- |
| **PLATFORM** | **High Resource Usage Intensity** | **Resemblance b/w the requirements and the platform** | **Familiarity with the Platform** |
| Web Application | No | Yes | Yes |
| Desktop Application | Yes | No | Yes |
| Mobile Application | Yes | Yes | Yes |

**4. BROWSER:  
  
Terms of reference:  
  
We are considering:**A. Internet Explorer  
B. Google Chrome  
C. Mozilla Firefox  
D. Safari   
 **The criteria to which we would use to decide which browser our software would target:**A. Familiarity with the browser  
B. The overall popularity of browser  
C. Compatibility between the browser and proposed softwareMuch like the operating system, there is little importance in the need for only one specific browser to be chosen. More relevant is the need to consider all of them, considering they all see relative use by stakeholders. Therefore, the developers would conduct cross-browser testing once the app is complete.

However, the recommended browser for this app would be As a web application, it’s reasonable that the mainstream web browsers would all be compatible. But for the developmental stage, the developers are using Google Chrome.

**Recommended Browser for the Web Application:** All of the above mentioned browsers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Browser** | **Familiar** | **Popular** | **Compatible** |
| Internet Explorer | Yes | Yes | Yes |
| Google Chrome | Yes | Yes | Yes |
| Mozilla Firefox | Yes | Yes | Yes |
| Safari | Yes | Yes | Yes |

**5. SERVER:**

**Terms of reference:**

**We are considering:**A. Local server

B. Renting a server

**The criteria to which we would use to decide on the server used:**

A. Whether the server is paid or free  
B. Whether the server is local or public

C. Ease of usage by the team members

None of the team members have experience with renting and using a server for a web app. Therefore, the team has decided to use a local server which would be created using Node.js to host the web app being created by the team.

|  |  |  |  |
| --- | --- | --- | --- |
| **Server** | **Paid/Free** | **Local/Public** | **Ease of use** |
| **Local server** | Free | Local | Yes |
| **Renting a server** | Paid | Public | No |

**RISK ANALYSIS PLAN**

The main areas that risks can have an impact on, in terms of software development, can be seen as:

* **Timeliness**

Delivering the software on time

* **Cost**

Delivering the software on a budget

* **Fulfilments of the requirements for the software**

The requirements can change, be misinterpreted, or simply be infeasible to the team designing the solution.

* **Usefulness**

The software can be delivered on time and budget, but not do what the product owner wanted it to do

**Cost**:

The cost factor doesn’t apply to the team’s project because the team would be hosting a local server instead of renting a server.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RISK** | **ESTIMATED IMPACT AND LIKELIHOOD** | **RISK MONITORING STRATEGY** | **RISK MITIGATION** | **RISK TYPE (TECHNICAL/ORGANISATIONAL)** |
| Communication with the client may not be sufficient, leading to requirement changes not being properly communicated by the client. | **Impact** - If the required changes aren't properly  communicated then this can result in the client not being happy with the final product or the final product being faulty.    **Likelihood** - Usually the chances of this happening is less as the team’s process model makes sure that no doubts in requirements would lead to a faulty software | Upon every discussion with the client, reiterate the related requirements to check that it is still agreed upon. | Communicate with the client and clarify the requirements. | Technical |
| **RISK** | **ESTIMATED IMPACT AND LIKELIHOOD** | **RISK MONITORING STRATEGY** | **RISK MITIGATION** | **RISK TYPE (TECHNICAL/ORGANISATIONAL)** |
| Communication difficulties with the client may lead to delays in task completion time. | **Impact** - The impact of this risk is low because most of the team members will follow a time schedule, which is made as per the requirements they currently understand    **Likelihood** - The likelihood of this happening is medium as some of the clients only get to meet the developers in pre-agreed meetings and usually don't have any other way of communicating with them other than through email | Try to ask all of the potential questions that may come up while discussing the requirements with the client. | Ensure that the team has some method of communicating with the client in ‘off-hours’ other than through email. An example of this could be through a social media group. | Organisational |
| Insufficient skills required for the task. | **Impact** - The impact of this risk is high as none of the team members have worked on a project like this before as a result of which if the team members don’t develop the skills needed for the project to succeed, then the end product would fail miserably    **Likelihood** - Likelihood of this happening is medium as we are still learning the language as we develop the program. | Discuss the skills that might be needed with the team members and work hard to develop those skills | As a result of discussion within the team, come up with a strategy that allows for programming tasks to be allocated according to experience. If nobody has the required knowledge, work on the specific task as a group in order to reduce time wasted. | Technical |
| **RISK** | **ESTIMATED IMPACT AND LIKELIHOOD** | **RISK MONITORING STRATEGY** | **RISK MITIGATION** | **RISK TYPE (TECHNICAL/ORGANISATIONAL)** |
| Team members have other matters to tend to. | **Impact** - The impact of this risk is medium as the absence of team members can result in the team not finishing the requirements on time.    **Likelihood** - This risk is likely to happen as the commitments of the developers for things other than this project might might affect their ability to work. | As soon as any team member feels like he/she cannot complete a task due to other commitments, the member should inform the rest of the team members so that the team doesn't fall behind on their schedule | Allocate time responsibly towards all assignments so as to ensure a lower pressure environment. Also, allow a margin of error for completion of each task. | Organisational |
| An assigned task not be completed on time by a member. | **Impact** - If the assigned work is not completed by a team member it can lead to an increase in completion time or failure to wrap-up the requirements on time    **Likelihood** - The likelihood is medium as sometimes team members do not use their time wisely or get distracted by other things or lack the skill set required. | Every member discusses their progress on the assigned task regularly (every 2 days via Trello or Facebook Messenger) | Question the group’s progress at meetings. Ensure that anyone who is struggling is offered help. | Technical/Organisational |
| **RISK** | **ESTIMATED IMPACT AND LIKELIHOOD** | **RISK MONITORING STRATEGY** | **RISK MITIGATION** | **RISK TYPE (TECHNICAL/ORGANISATIONAL)** |
| Connection issues in communicate and file submission | **Impact** - The impact of this risk is low as  If a particular network isn’t working then it is not hard to find another one.    **Likelihood** -The likelihood of this happening is quite low, the reason being that as Monash students we have access to the networks on campus as well as our personal ones. | Let the lecturer, demonstrator know about the issue as soon as possible | If struggling with a certain network, try to find another one for the purposes of submitting the files. This could mean using the university’s network if a team member is experiencing connectivity issues at home. | Technical |

**INDIVIDUAL CONTRIBUTION**

|  |  |
| --- | --- |
| **TEAM MEMBER** | **CONTRIBUTION** |
| Vibhas Kamal | * Got together with the team on 4 occasions and worked on the Project Plan, Risk Analysis Plan and Analysis of Alternatives |
| Isara Arunanondchai | * Got together with the team on 4 occasions and worked on the Project Plan, Risk Analysis Plan and Analysis of Alternatives |
| Jack Beard | * Got together with the team on 4 occasions and worked on the Project Plan, Risk Analysis Plan and Analysis of Alternatives |
| Manvendra Singh | * Got together with the team on 4 occasions and worked on the Project Plan, Risk Analysis Plan and Analysis of Alternatives |
| Quang Ly | * Got together with the team on 4 occasions and worked on the Project Plan, Risk Analysis Plan and Analysis of Alternatives |