CS261: Requirements Report

1 Team Organisation

1.1 Roles

For a team of 6 people, we have assigned a total of 3 unique roles: project manager, research/business analytics, and software developers. The breakdown for each individual is as follows:

- Project Manager - Andreea Nicolae

- Research/Business Analytics - Raihanah Lukman

- Software Engineer - Heath Nicholson, Aris Papakonstantinou, Billy Pentney, Cem Yilmaz

We have further broken down the software engineering team to a total of 3 back end and 1 front end member. The front end software engineer was determined to be Aris Papakonstantinou.

1.2 Methodology

Important decisions with regards to the project are generally taken democratically to ensure the engagement and agreement of most if not all team members. Our main methodology utilised are scrums. Scrum was found to be the most suitable: the customer only wants a single delivery, the scope is unclear and timescale is unclear, and we all undergraduate students, meaning that we have little project experience. Our scrum master was assigned to be Andreea Nicolae. Meetings were democratically agreed to be organised with a minimum of set 2 meetings a week, every Friday (in-person) and Monday (online). We have also agreed upon having small or urgent meeting opportunities when required.

2 Functional Requirements

2.1 Tracking

The tracking of our software will be conducted through linkings to a repository from GitHub and semi-frequent surveys. (will the user link their account or the public repository?). Specifically, GitHub will enable us to track multiple useful statistics that will aid us with risk assessment of the project, e.g., code, commit data with dates i.e., deadlines, how regularly the project is updated, bug reports and tags. Surveys will help us analyse trends of soft metrics of a project. The manager is further able to edit any input hard metric data later in the future if any changes in the plan are present. The tracked data and evaluated riskiness will be output through the website only through manager's account, presented in the website using graphs and percentages of factors such as budget, time, completion etc. The outputs themselves, we have decided, will be different depending on each person's role in the project, based upon a role-based hierarchy. Possible roles include, but are not limited to: software engineer, project manager, back end lead, front end lead, etc.

Democratically, we have decided that only the manager will have the permission to view the team's statistics and riskiness. Sharing the riskiness of the project with the client may prove to be disadvantageous for the project i.e., the client may decide on cancelling the project after the risk assessment. We believe that this decision is exclusive to the project management team. Furthermore, sharing the progress of the riskiness of the project with the non-senior team may prove to be demotivating or a source of low team morale if it is found that the project is risky. As such, this information will be limited to only certain members of the project. Additionally, a could have feature would be the output of the riskiness and diagrams to be in a saveable pdf output for the manager so they can share or print the output externally.

2.2 Metrics

We have split the definition of metrics into two: hard metrics and soft metrics. Hard metrics mainly concern of factors that are measurable directly and is the initial input of the project into our system. These include, but are not limited to factors such as budget, experience of employees, code in GitHub, deadlines, etc. Soft metrics has been defined to be the each team member's view, morale and motivation of the project.

It is vital that we gain data of these metrics from the participants of the project, especially of soft metrics (citation here). We have decided that the method of gaining hard metrics would primarily be through the initial risk assessment. However, the soft metrics would require to be measured in a regular time through surveys of all members. The survey would be done through their account in our website at a regular basis. This "regular basis" can be determined by the project manager, with a suggested time frame. Consequently, we have also highlighted the idea that such a survey will be short to minimise burden of the members of the project. Lastly, whilst it is not necessary that every team member fills the survey, the larger the sample size of the participants, the more accurate the risk assessment presented. It is recommended by our software that the majority of the project participants fill in the survey. Otherwise, the project would be warned to be at risk due to lack of engagement, which is an important factor of success (citation here) of a project.

A comprehensive list of hard metrics and soft metrics has been determined within our team. The hard metrics are as follows:

- **Budget** - the initial capital investment onto the project is important as it can determine whether it is sufficient and can be compared to trends that are proven to be successful from our learning algorithm to then output correct suggestions i.e., what budget really is required according to team size and expertise.

- **Complexity** - complexity of a project is a necessary measurement, as it can determine the likelihood of success. A highly complex project can be suggested to be broken down to less complex parts for digestion and ease of work.

- **Expertise** - this hard metric is necessary to ensure that technical know-how is sufficient depending on the complexity of the project. This is measured when a user is signed up e.g., years of experience within their related field of the project. If an expertise of a project is found to be critical, it will be suggested that the project hires new employees with required amount of expertise, whilst maintaining an optimal team size.

- **Team size** - a hard metric that aid us in measuring workflow and scale of operation, where there machine will give suggestions according to learned economies of scale. This will be measured by the number of registered members for a specific project. Too many members would require to some to be dismissed from the project, too little would be that new members are required, all according to already existing data of expertise.

- **Time frame** - we are required to know if the project is able to deliver according to their own deadlines, so that the riskiness can be adjusted accordingly. This can be addressed by re-adjusting the existing time frames accordingly to the trends of that project.

- **Meeting requirements** - we are also required to know if the project is able to deliver all things that are requested from the customer.

- **Code** - code is an essential bit of a software project, can be used to determine the quality of the project. Poor code and code management can result the failure of a project. Poor code can be caused by many factors e.g., lack of documentation, which will be analysed and suggested accordingly.

The soft metrics have been determined as follows:

-

2.3 Riskiness

The riskiness of the project will be evaluated through an initial input of data done by the project manager, and further, it is necessary that most of it not all project members sign up to the project using our website to maximise the accuracy of the risk assessment. The initial input will consist of hard metrics such as the budget, estimated time frames, team members, topics related to development i.e., tools/techniques in use and methodology (how do we measure topics?). The system will further keep track of the project's phase and soft metrics in each phase e.g.,

planning, concept, design, implementation, deployment, etc. as factors of risk are different for each phase (citation). This will be done through user input, e.g., surveys. Technical updates will be done through the linkage to GitHub repositories. Risk calculation will be done using through the use of most hard metrics and soft metrics.

It is further essential that input can be taken throughout the project to edit any changes to any of the hard metrics. We have decided that the project's manager can adjust the project's hard metrics such as the time frame, budget, etc. The risk assessment will be adjusted accordingly after each update or a specific time frame (why not have an update button?). With such updates, the back end algorithm will analyse the trends and output a risk assessment accordingly with the new data.

2.4 Use of Existing Data

The initial model will be initialised using data that is provided by research. When a project is marked complete, our system will prompt the manager to input a final summary. The final summary consists of the following:

- Rate perceived degree of success

- Total cost

- Time of completion

We could further prompt for further updates during project lifetime e.g., 3 month reviews. Lastly, the back end of our system has been agreed to consist of unsupervised learning. The result of project fed back into artificial intelligence (AI). Depending on the outcome of the project, the AI will adjust the weightings of factors and metrics that is used in risk calculation.

Initially, the project will be filled with existing public data of successful and failed projects. Using the existing data, the algorithm will be able to analyse and differentiate what makes up of a successful project through its trends, and give suggestions accordingly for hard metrics. Hard metrics mainly consist of numerical data, therefore, their suggestions will be straightforward and analysable i.e., number changing. For soft metrics that mainly consist of social problems, existing analysis and data of suggestions exists in [The Book] in case of a detected low trend.

---

3 Non-Functional Requirements

3.1 User Interface

The software in our project will be minimal by design i.e., limited colour palette, minimal text, easy to navigate, as it was requested that the system is "suitable for non-technical users" and "should be intuitive and require little to no training". Consequently, our website will consist of well-labeled user interface with little user interaction other than clicking and entering text. Furthermore, there is a potential for implementation of an FAQ or even video tutorials on the usage, if found suitable.

3.2 Technicalities

3.3 Account Management

Our system will incorporate a simple account management system. Users will be able to create, login, logout and delete a local password-protected account with our software. Each user is able to create any number of project profiles, each with a title, team, deadlines, keywords, data on hard metrics and methodology. Creator of a project can may further grant access to other users in the system with permissions such as read, write, full access. Only people with full access can view the project's riskiness and suggestions. A user can only view projects which they have read access to.

Furthermore, each project can be linked to an online repository such as GitHub. The project status such as bug reports, commits are stored locally and synchronised with the repository when an internet connection becomes available. Project status, on the other hand, can be updated via our software if and only if permissible and viewed as a graph describing changes against time.