# Risk Assessment and Management Plan (RMP)

The purpose of a risk assessment and risk management plan is to preemptively identify, analyze, and address the various potential risks inherent in our project. These risks may manifest in different types, such as technical or management risks. They also exhibit variations in risk scores, determined by comparing the impact and probability of each risk (refer to Figure 1). The Risk Management Plan (RMP) holds significant importance in any software project as it empowers teams to foresee challenges and explore potential solutions to issues that could arise, ultimately impacting the project's success.

Risks were identified during brainstorming meetings among as many teammates as possible. This was to get as many raw ideas as possible and to identify all possible risks with less chance of any slipping through the cracks, or personal bias limiting the scope of our assessment. Once we had all this raw data, a dedicated team of 3 students met to go through the risks and elaborate on them, as well as assign them impact and probability ratings from low to high. This led to the creation of the tables below.

When looking at Impact and Probability, we use these estimates to assess risks:

| | | | |
|---|---|---|---|
| Impact | 1-3 days | 3-7 days | 7+ days |
| Probability | 0–5% | 5–15% | 15+% |

These are rough estimates of course, but they help us categorize our risks and organize discussions regarding where to place each risk in the table below. Without these concrete numbers, two people might end up with too much variance in their assessments of a risk even if in reality, they were thinking of the same risk categories.

| Impact<br>Probability | Low | Medium | High |
|---|---|---|---|
| Low | 1.4<br>1.6<br>1.8 | 1.9<br>1.12<br>1.14 1.17 | 1.3 1.5 1.13<br>1.15 1.16 1.19 |
| Medium | | 1.7<br>1.10 | 1.11<br>1.18 |
| High | | 1.1<br>1.2 | |

Figure [1]: Risk management chart

| Risk ID | Risk Type and Description | Risk Score | Resolved in Sprint | Strategy and Effectiveness |
|---|---|---|---|---|
| US-1.1 | ● Technical<br>● Management<br>● External<br>● Budget<br>● Schedule<br>● Etc. | ● Low<br>● Medium<br>● High | | ● Mitigate<br>● Accept<br>● Avoid<br>● Transfer |
| T.1.1 | **Technical**<br>*Integration challenges between app and website:* The website front-end may not be completely responsive and the components developed in the website may not translate well in the app | High probability<br>Medium impact | Sprint 2 | **Consequence** is a reduced user experience for mobile<br>**Avoid** through constant cross-platform and different screen size testing and using react/react native, a framework that facilitates code reuse |
| T.1.2 | **Technical**<br>*Unfamiliarity with the tech stack for front-end and back-end integration:* The teams unfamiliarity with the tech stack used can lead to integration difficulties | High probability<br>Medium Impact | Sprint 2 | **Consequence** is a higher proportion of bugs and issues in the finished product, as well as a slower development time<br>**Mitigation** through training sessions for team members on the elements that are lesser known technologies, utilize team members strength and previous experiences, frequent meetings so there is coherence between front and back team, so collaboration is seamless. |
| T.1.3 | **Technical**<br>*Inconsistent data handling for different user types:* There are many different types of users with | High impact<br>Low probability | Sprint 2 | **Consequence** is data errors and thus bugs<br>**Avoidance** through making sure that the data handling |

| | | | | |
|---|---|---|---|---|
| | different requirements and views of the platform, this could result in inconsistencies and potential data errors | | | protocols are clear and data validation checks are made through the development process |
| T.1.4 | **Technical**<br>*Dependence on external APIs:*<br>Dependencies on external APIs for certain features may introduce risks, including changes in API specifications or unexpected downtime. | Low probability<br>Low impact | Sprint 2 | **Consequence** is security risks and dependency on API changes<br>**Accept**, we have no control on external software and have no contact with API providers |
| T.1.5 | **Technical**<br>*Insufficient documentation* for the system architecture can lead to data redundancy, lack of understanding and inconsistent updates when implementing the database | High impact<br>Low probability | Sprint 1, Sprint 2 | **Consequence** is lower overall code quality and poor understanding by new programmers on the project<br>**Avoid** by working on the domain model early on and ensuring collaboration between the development team and the team members working on the system architecture |
| T.1.6 | **Management**<br>*Disorganized teams* leading to double work. Multiple teams/members may work on the same features or systems and overwrite each other's contributions. As well as waste time in the process. | Low probability<br>Low impact | Sprint 1, Sprint 2 | **Consequence** is time lost and possible conflict to decide which version is "better"<br>**Mitigate** by having team leaders communicating frequently, a Github repository so all members are aware of current work, and task management software such as Trello to organize tasks. |
| T.1.7 | **Management**<br>*Not meeting deadlines* or meeting internal goals. Hard deadlines may be missed, disappointing the client and disrupting the overall development timeline | Medium probability<br>Medium impact | Sprint 2 | **Consequence** is disappointing the client, as well as breaching contractual obligations.<br>**Mitigate** by creating internal "soft" deadlines to be respected by the team, as well as encouraging |

| | | | | |
|---|---|---|---|---|
| | | | | communication and cooperation. |
| T.1.8 | **Budget**<br>*Limited budge*t for tooling can present challenges since this is for a course. | Low probability<br>Low impact | TBA | **Consequence** is having limited options for hosting and/or tools<br>**Mitigate** by using free trials and free versions of tools. Some companies offer free premium versions for students |
| T.1.9 | **Management**<br>*Unavailability* of team members due to unforeseen circumstances can impact project progress | Low probability<br>Medium impact | Sprint 2 | **Consequence** is slowing development due to dependency on certain members<br>**Transfer**<br>Cross-training team members to be able to perform a variety of tasks and outsourcing tasks when necessary |
| T.1.10 | **Technical**<br>*Insufficient user testing* may lead to undetected bugs or usability issues | Medium Probability<br>Medium Impact | Sprint 2 | **Consequence** is bugs and usability issues are discovered by users, impacting their experience negatively<br>**Accept**<br>It is an impractical goal to want to ensure with certainty that there are no bugs in the system. Our time and resources being limited, we will not be able to cover all bases. We still need to ensure an adequate amount of testing. We aim for 80% coverage. |
| T.1.11 | **External**<br>*Hacking into the mobile or web application*, allowing for abuse of the registration, financial, or reservation systems. | Medium Probability<br>High Impact | Sprint 2 | **Consequence** is data breaches and abuse by a third party<br>**Mitigate** by employing responsible coding practices and keep security in mind when designing the system. |
| T.1.12 | **Schedule**<br>Team may face *challenges in conducting regular scrum meetings* | Low Probability<br>Medium Impact | Sprint 2 | **Consequence** is a lack of team cohesion and confusion. |

|  |  |  |  |  |
|---|---|---|---|---|
|  |  |  |  | **Mitigation** by holding virtual meetings, utilizing when to meet, making meeting minutes so team members can stay updated and establishing clear meeting times to maximize attendance. |
| T.1.13 | **Technical**<br>*Neglecting core features* or requirements. | Low Probability<br>High Impact | Sprint 1 | **Consequence** is not meeting client/contractual expectations<br>**Avoid** by generating extensive user stories and thorough dissection of client needs. |
| T.1.14 | **Management**<br>*Resource Availability*: Potential shortfall in tools and programmers. | Low Probability<br>Medium Impact | Sprint 2 | **Consequence** is starvation for important tasks<br>**Mitigation**<br>Resource planning, cross-training and contingency reserves. |
| T.1.15 | **Management**<br>*Lack of stakeholder engagement* in the project. | Low probability<br>High Impact | Sprint 2 | **Consequence** is a disconnect between stakeholder wants and developer assumptions<br>**Mitigation**<br>Regular update meetings, stakeholder management strategies. |
| T.1.16 | **Technical**<br>*Non-adherence to UI/UX standards*: Failure to meet user interface design norms. | Low probability<br>High impact | Sprint 2 | **Consequence** is deprecating the user experience<br>**Mitigation**<br>Following UI/UX best practices , user testing. |
| T.1.17 | **Technical/Management**<br>*Misaligned code management practices:* Inconsistent use of version control and coding workflows. | Low probability<br>Medium Impact | Sprint 2 | **Consequence** is lower overall code quality and poor understanding by new programmers on the project<br>**Mitigation**<br>Mandatory code review policies. |
| T.1.18 | **Technical**<br>*Requirements and user stories backlog mismanagement:* | Medium probability<br>High impact | Sprint 2 | **Consequence** is possibility of missing requirements, as well as a bad priority system of requirements |

| | | | | |
|---|---|---|---|---|
| | Ineffective tracking and prioritization of project tasks. | 🟥 | | **Mitigation**<br>Use backlog grooming sessions to keep the backlog updated and prioritized |
| T.1.19 | **Technical**<br>*Design pattern misuse:* Incorrect applications of the design patterns. | Low probability<br>High Impact | Sprint 2 | **Consequence** is a worsened user experience<br>**Mitigation**<br>Conduct design review sessions. |

Table [2]: List of identified risks