Hazard Analysis
Flow

Team 9, min-cut
Ethan Patterson
Hussain Muhammed
Jeffrey Doan
Kevin Zhu
Chengze Zhao

Table 1: Revision History

Date	**Developer(s)**	**Change**
Date1	Name(s)	Description of changes
Date2	Name(s)	Description of changes
...	...	...

Contents

[You are free to modify this template. —SS]

1 Introduction

[You can include your definition of what a hazard is here. —SS]

2 Scope and Purpose of Hazard Analysis

[You should say what **loss** could be incurred because of the hazards. —SS]

3 System Boundaries and Components

[Dividing the system into components will help you brainstorm the hazards. You shouldn't do a full design of the components, just get a feel for the major ones. For projects that involve hardware, the components will typically include each individual piece of hardware. If your software will have a database, or an important library, these are also potential components. —SS]

4 Critical Assumptions

[These assumptions that are made about the software or system. You should minimize the number of assumptions that remove potential hazards. For instance, you could assume a part will never fail, but it is generally better to include this potential failure mode. —SS]

5 Failure Mode and Effect Analysis

[Include your FMEA table here. This is the most important part of this document. —SS] [The safety requirements in the table do not have to have the prefix SR. The most important thing is to show traceability to your SRS. You might trace to requirements you have already written, or you might need to add new requirements. —SS] [If no safety requirement can be devised, other mitigation strategies can be entered in the table, including strategies involving providing additional documentation, and/or test cases. —SS]

6 Safety and Security Requirements

[Newly discovered requirements. These should also be added to the SRS. (A rationale design process how and why to fake it.) —SS]

7 Roadmap

[Which safety requirements will be implemented as part of the capstone timeline? Which requirements will be implemented in the future? —SS]

Appendix — Reflection

[Not required for CAS 741 —SS]

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. What went well while writing this deliverable?
    - Chengze - I found that our team collaborated effectively in dividing the work and keeping our analysis consistent across sections. We communicated clearly about potential hazards and quickly agreed on the scope and structure of the document. Using LaTeX templates also helped maintain a clean and professional format throughout.
    - Ethan - I think something that went well in this deliverable was the team collaboration. Everyone communicated well and we tried our best to have PRs up and a proper review process. We learned from our last deliverable that saving review to the last minute makes it really hard to actually incorporate the feedback, but it was better in this milestone.
    - Hussain -
    - Jeffrey -
    - Kevin -
2. What pain points did you experience during this deliverable, and how did you resolve them?
    - Chengze - A main difficulty was completing some parts of the hazard analysis without direct communication among team members, since it is closely related to the SRS content. Some sections depended on finalized requirements or component details that were still being developed. We resolved this by holding short sync meetings and clarifying assumptions so our analysis stayed consistent with the SRS.
    - Ethan - It was difficult to think about hazards related to our product, since our project is purely software and doesn't really have any potential physical harm to users or the environment. We resolved this by working as a team and brainstorming other hazard areas that we wouldn't typically think of at first.

- Hussain -
- Jeffrey -
- Kevin -

3. Which of your listed risks had your team thought of before this deliverable, and which did you think of while doing this deliverable? For the latter ones (ones you thought of while doing the Hazard Analysis), how did they come about?

    - Our team had already identified general risks such as data loss and performance issues early in the SRS phase. However, while doing this deliverable, we recognized new risks related to user input handling and rendering consistency in the canvas editor. These came up as we analyzed possible failure modes and realized how improper error handling or unexpected user actions could lead to crashes or corrupted note files.

4. Other than the risk of physical harm (some projects may not have any appreciable risks of this form), list at least 2 other types of risk in software products. Why are they important to consider?

    - Some important types of risk in software products are data security risks and usability risks. Data security risks, such as unauthorized access or data corruption, can lead to loss of user trust and legal issues. Usability risks, like confusing interfaces or unclear workflows, can reduce user satisfaction and prevent adoption of the software. Considering these risks early helps ensure both system reliability and a positive user experience. Another risk may be financial risk, where project delays or cost overruns could impact the viability of the product. If the project runs out of budget or takes too long to deliver, it may not be feasible to continue development or support.