Hazard Analysis
Sandlot

Team 29
Nicholas Fabugais-Inaba
Casra Ghazanfari
Alex Verity
Jung Woo Lee

Table 1: Revision History

Date	**Developer(s)**	**Change**
Oct. 21, 2024	NFI, JL, AV, CG	TA Feedback
Oct. 23, 2024	NFI, JL, AV, CG	Rev0
...	...	...

Contents

1 Introduction

Sandlot is intended to be the successor for the current McMaster GSA softball league scheduling and management platform. This project will implement the current functionality from the existing system including additional features such as a new login system, commissioner admin privileges, and an improvement to the robustness, ease of use, and maintainability to the platform.

Hazards are conditions of the product system that could lead to undesirable outcomes. In Sandlot, hazards of security, malfunctions, usability, among others will be analyzed.

2 Scope and Purpose of Hazard Analysis

Hazard analysis aims to identify potential hazards that may exist with the product, understand their causes and effects, and provide proactive strategies to mitigate them. The purpose of this document is to perform this analysis for Sandlot, investigate potential hazards and give means of mitigating the issues identified. Hazards to Sandlot could take the form of user interface misinterpretations, misinputted data, and database errors, leading to possible malfunctions with the product and/or user dissatisfaction. This document will first identify the system components and boundaries, state the critical assumptions, use a Failure Mode and Effects Analysis (FMEA) table to identify the hazards and their causes, effects, and mitigation strategies, and lastly the requirements that are discovered from the analysis.

3 System Boundaries and Components

3.1 System Components

The system can be divided into the following components:

- Authentication

  This component authenticates user inputted login information with the database and determines whether a login is successful.

- Scheduling

  This component generates a schedule based on user inputted availability data. Additionally, it allows for rescheduling events in these schedules after they have been generated.

- Accounts

  This component allows users to create, delete or edit accounts. Additionally, account types determine the ways in which a user can interact with the system, its data, and its components.

- Team Structure

  This component is a data structure of the system which allows accounts to be grouped together such that a group of accounts interact with the system, its data, and its components in the same way as each other. Additionally, accounts can be added/removed from teams, users can request to join a certain team, and teams can be created/deleted.

- Scoring/Standings

  This component allows users to input score data associated with specific events and will generate cumulative standings based on score data.

- Alerts

  This component alerts users of important information related to the system.

- Database

  This component stores the system's data.

- User Interface

  This component displays the system's data to the user.

3.2 System Boundaries

The system limits itself to the components mentioned before; the front end and backend of the website. External APIs that connect to our system are considered as outside the system boundaries. This would include the webserver, and weather tracker, among others.

4 Critical Assumptions

- It is assumed that users will not attempt to maliciously attack the system.

5 Failure Mode and Effect Analysis

Component	Failure Mode	Failure Effect	Failure Cause	SR	Recommended Actions
Authentication	User login information does not match database stored login information.	User cannot access their account.	Misinputted user login information.	SR-7	Misinputted user login information should give a warning to the user when login information does not exist or is incorrect.
Scheduling	Captain inputs incorrect availability dates at start of season	Schedule is generated incorrectly.	Captain's user error.	SR-1	Allow captain to resubmit availability that overwrites previously submitted availability if submitted before the due date.
	User reads schedule incorrectly.	User may not attend a game by accident or travel to a game at the wrong time.	Schedule is unclear, unreadable, or user makes an error.	AC-1 AC-2	Ensure schedule data and structure is visible and readable.
	Teams availability data have scheduling conflicts.	System cannot make a valid schedule.	Availability data has scheduling conflicts.	SR-8	Teams will be given a warning if their availability data has scheduling conflicts.
	Captain doesn't respond to another captain's reschedule request.	Captain who sent the reschedule request will not know if the opposing captain wants to reschedule or not.	System doesn't adequately notify the opposing captain about the reschedule request.	SR-5	Make sure the captain who needs to accept the request cannot avoid seeing the request notification.
Teams	User is assigned to or joins the wrong team.	User will be shown the wrong schedule and team information.	Interface inputs are unclear or user makes an error.	AP-1 AP-2	Ensure that when assigning a team for a user, the team options to select from are clear and the input section is readable.
Accounts	Account is given the wrong permissions.	User has access to actions they shouldn't use.	Interface lacks security or user makes an error.	SR-2	Giving an account additional permissions should give a warning to the user when configuring accounts.
	All commissioner level accounts are deleted.	There are no accounts left that can give commissioner level permissions.	System allows for all commissioner accounts to be deleted.	SR-3	Do not let users delete a commissioner account if there is only one left.
	User logs in on an account that isn't theirs.	User may make actions that are not desired by the account's owner.	Account owners are not careful with their login data.	SR-10	Remind users to keep their passwords secret andsecure.
Scoring/ Standings	Captain inputs the wrong score.	Standings and database has incorrect scores.	User inputs the wrong score.	SR-4	Allow the opposing team's captain to verify the score and contest it if the score is wrong.
Alerts	User reads alert incorrectly.	User may travel to a game that was postponed or cancelled, or miss out on critical information.	Alert is unclear, unreadable, or user makes an error.	SR-9	Ensure alert message is visible and readable.

6 Safety and Security Requirements

Requirement #: **SR-1**

Description: **If the season start availability due date hasn't been reached, a captain should be able to resubmit availability data which overwrites previous availability data.**

Rationale: **If the captain makes an error when submitting availability data they should be able to fix their error.**

Originator: **Alex Verity**

Fit Criterion: **If the captain has submitted data, they shall be able to overwrite it with new data.**

Customer Satisfaction: **4** Customer Dissatisfaction: **3**

History: **Created 2024-10-15**

Requirement #: **SR-2**

Description: **Giving permissions to users must be accompanied by an error that warns the user of the severity of the action.**

Rationale: **Accidentally giving permissions to users who shouldn't have them could result in unexpected errors.**

Originator: **Alex Verity**

Fit Criterion: **If permissions are being changed, a warning shall be displayed to the user before updating the permissions.**

Customer Satisfaction: **5** Customer Dissatisfaction: **5**

History: **Created 2024-10-15**

Requirement #: **SR-3**

Description: **If there is only one commissioner level account, that account cannot be deleted.**

Rationale: **Deleting this account would stop any more commissioner level accounts from being created, soft-locking the system.**

Originator: **Alex Verity**

Fit Criterion: **If there is only one commissioner level account it shall not be deleted.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-15**

Requirement #: **SR-4**

Description: **All match scores must be visible and contestable by other captains once recorded.**

Rationale: **Match score correctness is extremely important for a competitive league, if a score is wrong, captains should be able to request that it be fixed.**

Originator: **Alex Verity**

Fit Criterion: **Any match scores shall have the option to be viewed and contested by captains.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-15**

Requirement #: **SR-5**

Description: **Captains are adequately notified when they receive an alert or a reschedule request.**

Rationale: **Important information may be shared in alerts, and reschedule requests that go unanswered may be frustrating for captains. Notifications should reach their intended targets if possible.**

Originator: **Alex Verity**

Fit Criterion: **Assuming the user hasn't interfered, alerts and reschedule requests shall always be sent to a place the user receives notifications.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-15**

Requirement #: **SR-6**

Description: **Users can only access an account with correctly inputted login information that matches the database stored login information.**

Rationale: **Login information stored in the database should correspond to a specific account. The account should only be accessed by the correctly inputted login information details.**

Originator: **Nicholas Fabugais-Inaba**

Fit Criterion: **Assuming the user has correctly inputted the login details for an account stored in the database, they should be granted access into the corresponding account.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-25**

Requirement #: **SR-7**

Description: **Misinputted user login information shall provide a warning to the user, if login information does not exist or does not match the database stored login information.**

Rationale: **Login information stored in the database should match the user inputted login information. Feedback should be provided for the user to understand an error has occured when accessing an account with incorrect login details.**

Originator: **Nicholas Fabugais-Inaba**

Fit Criterion: **Assuming the user has misinputted the login details for an account stored in the database, they should be given a warning that notifies them about the login information being incorrect or not existing in the database.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-25**

Requirement #: **SR-8**

Description: **Teams will be given a warning if their availability data has scheduling conflicts.**

Rationale: **The system should be able to create a valid schedule, in which teams do not have conflicting availability data that schedules games for the same dates and times as other teams.**

Originator: **Nicholas Fabugais-Inaba**

Fit Criterion: **Teams should receive a warning about their availability data conflicting on the schedule.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-25**

Requirement #: **SR-9**

Description: **Alerts sent to users must be visible and readable.**

Rationale: **User may travel to a game that was postponed or cancelled, or miss out on critical information.**

Originator: **Nicholas Fabugais-Inaba**

Fit Criterion: **User receives an alert that is readable and clear enough for them to understand.**

Customer Satisfaction: **3** Customer Dissatisfaction: **3**

History: **Created 2024-10-25**

Requirement #: **SR-10**

Description: **Users are reminded to keep their passwords secure.**

Rationale: **If a user accesses another user's account they may do actions without the account owner's permission or that the account owner is unaware of.**

Originator: **Alex Verity**

Fit Criterion: **A reminder telling users to keep their passwords secret and secure is displayed to the user.**

Customer Satisfaction: **1** Customer Dissatisfaction: **1**

History: **Created 2024-10-25**

7 Roadmap

The hazard analysis resulted in new safety and security requirements being introduced to the project's already existing requirements located in the SRS. Provided the time constraints and current project deadlines, critical requirements will be initially addressed with lower priority requirements being investigated in the future. The hazard analysis will be used as a guideline to ensure the safety and security of the Sandlot platform's users.

Proof of Concept Demonstration (November 11-22):

- Authentication: SR-6 and SR-7

- Scheduling: SR-1, SRS(22), SR-8

Final Demonstration (Revision 1) (March 24-30):

- Scheduling: SR-5
- Teams: SRS(17)
- Accounts: SR-2, SR-3 and SR-10
- Scoring/Standings: SR-4
- Alerts: SR-9

Appendix — Reflection

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?
2. What pain points did you experience during this deliverable, and how did you resolve them?
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?
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?

Nicholas Fabugais-Inaba – Reflection

1. What went well while writing this deliverable?

   When writing this deliverable, outlining the different failure modes and effects helped in understanding what hazards could impact our project from more than just a surface level view. More specifically, content in components such as scheduling and alerts, aided in my understanding of how each user may be impacted by a type of failure that could occur within the system.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   Some of the pain points from this deliverable had to deal with brainstorming appropriate hazards that may impact a user in the system. With many working components, there are a lot of hazards that could occur or that could be easily handled. Understanding which hazards could impact the users the most, which the team would then focus on to address primarily

before other secondary hazards, was really crucial in the completion of this deliverable. These pain points were resolved by brainstorming together as a team our own failure modes that we would then review amongst each other, if a member needed a second opinion on the component they were working on. This would help all of us to understand if the hazard was crucial to address for the sake of the system users or if it could be dealt with in an alternative way, and therefore not need to be listed.

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?

   Before this deliverable, one of the bigger risks our team had thought of before this deliverable was what we would need to do to address scheduling conflicts. We knew the scheduling portion of the system was an important aspect of this project and would need to be dealt with appropriately to have a robust system rather than the current platform that is prone to many errors. Some of the latter risks such as the alerts or the accounts risks came about as the team understood alerts could be misread, so to make it as easy as possible for users, the platform would need to make sure any announcements were to be clear and readable. Additionally, for account risks, on a rare occasion where all commissioner level accounts are deleted, we thought of making sure to put a safeguard so at the very least there could be one account, since users could make the mistake of deleting accounts by accident.

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?

   In software products, 2 other risks could be associated with a system shutdown from loss of battery life or some type of malware that could impact the software in a negative way. These would be important to consider as in the case for a failure from the system, there should be a failsafe to deal with such a mode as system integrity could be impacted as well as the user's device itself. In terms of malware, it could also impact the system's integrity leaving it vulnerable to impact users with faulty links that could damage a user's device. Malware could also crash the software system itself, which would be detrimental.

Casra Ghazanfari – Reflection

1. What went well while writing this deliverable?

   Writing the "System Components" section of this deliverable went well for our team. This is because going into this deliverable we already had

a concrete idea of the structure of our overall system. This allowed us to quickly partition the system into components based on our ideas for the overall structure. Moreover, due to our existing understanding of the overall system we were able to easily envision and determine how the derived components of the system would interact with each other and the system boundaries.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   The biggest pain point during this deliverable was connecting hazards in our FMEA table to security requirements (SRs). This is because our SRS deliverable was incomplete at the time of writing this deliverable and did not have a proper set of security requirements to connect to any of the hazards identified in the FMEA. This significantly increased the amount of work needed for us to properly complete this deliverable and generally slowed down the process of creating the FMEA table as well.

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?

   The team didn't concretely identify any risks prior to working on this deliverable, but had loose ideas of more obvious potential risks to include in the deliverable. "The user incorrectly inputting their login information" or "the user joining the wrong team" are examples of risks we had thought of before working on this deliverable. Risks like "all commissioner accounts are deleted" and "account is given wrong permissions" are examples of risks we thought of while working through this deliverable. Generally, the idea to include these risks came from carefully evaluating each individual component defined previously in the document and asking ourselves the question: "if anything could go wrong, what would go wrong?". From this line of thinking we were able to envision and identify uncommon risk scenarios for the system that were tied to the system's more complex components.

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?

   (a) Security risks
   Software products that handle sensitive data (i.e. personal / financial / health information) must consider the risks of handling sensitive information. Data can be stolen through unauthorized access and

data breaches and can cause financial and reputational harm to both the users and owners of the product. Therefore, security risks such as data breaches are important to consider when handling sensitive data, especially when trying to comply with data protection regulations.

(b) Operational risks
Software products that require high availability must consider the risks of relying on services outside the product's system boundary. For example, when using a hosting service provided by another organization to host the product's website or database, There's the inherent risk that if the organization hosting the service faces issues, so will your product. Issues like these can cause delays or system downtime for your product which is especially harmful if the product requires high availability as downtime negatively impacts the user satisfaction and experience of users of the product.

Alexander Verity – Reflection

1. What went well while writing this deliverable?

   The new requirements added needed coverage to the SRS document. Thinking about hazards helped a lot in adding requirements, and I am much more confident about the requirements document after the hazard analysis. Once again the TA meeting was extremely helpful, many good questions answered insightfully. The TA meetings help team confidence above all else. The supervisor meeting also was very helpful and will lead to many other changes to the SRS.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   What to add as a critical assumption was something the team debated over a long time. I feel like we ended up with a good critical assumption section but it took a lot of thinking and it is probably the part of the document we are least confident. The system boundary section was also difficult and required a lot of thinking.

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?

   We had thought of the coaches inputting incorrect scoring, as that is the part of the scoring process that is completely reliant on users to work correctly (however our supervisor has told us that the current system as never had an issue with scoring). Most of the risks we thought of while doing the deliverable. They mostly came about by analyzing the different parts of our system in section 3.

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?

   In our solution specifically, lack of fairness in a competitive competition. The league, although not every team needs to be of a high skill level, relies on teams taking it somewhat seriously for the most fun when playing games. Fairness is an important element of this. Without fairness players would be frustrated and likely not participate in the league. Another type of risk is privacy, with users personal information being leaked and used in ways the user didn't intend. This is important as privacy is a growing concern and something people value. It can also be financially dangerous or even physically dangerous, although that specifically is more rare.

Jung Woo Lee – Reflection

1. What went well while writing this deliverable?

   I think coming up with components went very smoothly. There was not much difficulty in figuring out the different sections of our system. As an added bonus, I found it very helpful to map out the future design of our system, and its major parts. This lets us begin to think about how these parts interact with the users and each other.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   There were significant issues with figuring out if certain issues counted as hazards. One example was about scheduling conflicts, and initially we had thought that two captains entering conflicting schedules at the same time should have been the hazard, however, further discussion made us realize this was becoming too implementation specific. After a long discussion, and even debating whether it was a hazard at all, we had come to an agreement on the current hazard in the FMEA table.

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?

   We had thought about the 'captains inputting wrong scores' and 'users joining the wrong team'. Most of the others we have come up with during the hazard analysis process. How these came about was as the team drew up the system components, we were more clearly able to see how users would interact with each component, or how components would function independently from other components. This allowed us to more readily

think of potential problems to the components and make our FMEA table. For example, 'all commissioner level accounts are deleted' was able to be thought of as we had thought to make 'Accounts' a component of our system. As we thought about how accounts could mess up somehow, it led us to think about the account hierarchy in our system, then to the hazard.

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?

   One kind of risk would be the risks associated with the project. I mean this as an umbrella that includes design, time, compliance, among others that occur during the creation of the software product. These are crucial to consider, as this can often make or break a product just as much as a bad product going to deployment. For example, a product that was made and misses the standards laid out for it would result in a large headache to go back and change, or a subpar system.

   Another type would be risks associated with the post-deployment era of a product. More so in terms of maintainability, adaptability, and portability. These can be important to consider if a product is slated to function for a long period of time, and receive adequate support for a long time period, it should be made as risk free as possible in these aspects.