CSE 360 Project Report Number 6
Team Tu37

Team Member Names:

1. Alma Babbitt
2. Ishan Yelnoorkar
3. Karryl Dumalag
4. Trevor Huss
5. Zachary Litwin

Table of Contents

Table of Contents

1. The Problem to be Addressed

1.1. Target Organizations

The organizations with which EffortLogger 2.0 will be most compatible are companies that have a large number of employees, a focus on efficiency and accountability, a data-driven approach to problem-solving, competent developers who are willing to back up their work, and managers who are adept at concluding individual employee data. Companies that have at least a few of these descriptors will surely find success in using EffortLogger 2.0. The software has been designed to complement the large-scale and data-driven companies that exist in the development world today.

1.2. The Problem

The customer has a system called EffortLogger. It is a set of tools that allow the customer to track necessary project progress and do accounting effectively. However, since the creation of the system 20 years ago, the firm has gotten larger, and the business requirements have grown significantly. The previous solution was an Excel-based solution created in Visual Basic that provided a set of Excel tabs that gave the appearance of an application interface. While the customer's leadership team is content with the current solution, a newer system is imminent to accommodate the changes in the business.

During the interview, we identified three stakeholders that contribute to the success of this project and the product. The three primary stakeholders are the customer, the user, and the supervisor. The customer dictates the needs of the firm and the vision for the newer system, while simultaneously the users dictate the efficiency of the product. Since the users are primarily dealing with the system to report performance data, they dictate what is required from the system. Finally, the supervisors are responsible for the firm's success, for which they dictate the tools required for them to efficiently and effectively supervise performance data.

Customer Problem

1. Employee privacy:

The customer holds employee privacy as the highest priority. However, added pressures have expected the customer to ask employees to provide details about productivity rates, defect rates, and other relevant data to resize the firm and reduce contingencies. Employees are concerned about improper use of personal performance data and hence prefer the Excel-based Effort Logger because it allows them better control over their data. However, the older system prevents the customer from getting automated performance data and synchronous cross-functional team projects and prevents the customer from maintaining the anonymity of their employees.

2. Confidential information security:

The leadership team is concerned about hackers accessing confidential information, including plans, schedules, budgets, and specifications. Therefore, security is of utmost importance. While planning and formulating project plans, timetables, and details, those who are involved in the management and formulation need to have a deep understanding of security risks and how to avoid them. Moreover, they also need to be adept at responding to security threats and the corresponding risk avoidance while retaining maintainability and integrity across all products.

3. Enterprise-scale support for agile and quality:

The customer has increased its solution output by a factor of 20 since the Excel-based EffortLogger was used. The upkeep has become larger, causing the customer to create a QA (quality assurance) engineering organization. Moreover, the previous system lacks the architectural and design support to enable future efforts to improve planning and managing tasks. The customer wants to move from the traditional scrum to a customized process based on enterprise-scale agile. Traditional scrum prevents the customer from dealing with size, complexity, and quality assurance.

User problems

Users (Employees) are responsible for logging their efforts. After an interview, here is a summary of a user’s problem while using EffortLogger.

1. Planning poker sessions tends to be long because team members will often spend time accessing personal repositories for data related to user stories. The previous system did not allow for criteria-specific planning of poker sessions, or it became tedious to put up historical data relevant to specific criteria.

2. The older system has a slower way of assigning weight to passing items during the initial screening. This reduces the ability to detect the significance of story points during the planning process.

3. The Excel-based effort logger does not support planning poker; therefore, some estimates significantly deviate from the team’s consensus. It lacks the ability for a "quick look," which would allow the employees to look at the contribution of each item.

4. The user also expressed that the old system is not very aesthetically pleasing and could use the ability to generate stylized poker cards with the ability to share them.

Supervisor problems

1. The firm works towards effort predictability and defect avoidance, which require anonymity; however, the previous system could not anonymize employee-identifying information.

2. A Proposed Solution

2.1. Solution Overview

The solution our team has derived has been focused on modernizing the current version of EffortLogger while solving its key problems. The main points our team has focused on consist of the following: creating a revamped user interface, ensuring employee privacy, securing confidential information, stability for large-scale Agile support, efficiency improvements, and planning poker support. These improvements have been initiated in the new version of EffortLogger and have been our main focus in this endeavor.

2.2. Major Solution Scenarios

Scenario 1. Large-Scale Agile Support

A company wishes to adopt EffortLogger as its main data source for employee performance. The company has thousands of employees who all wish to use this software, but they find that the software cannot handle the sheer amount of data being produced, which causes it to crash repeatedly.

Solution 1.

EffortLogger 2.0 has been created to support larger businesses through the use of SQL, which provides a much more solid database than the last V1.0 database.

Scenario 2. Planning Poker Support

An employee is annoyed at the fact that accessing software to perform a planning poker session is so difficult with the current version of EffortLogger. The employee wishes that EffortLogger could implement a seamless version of the planning poker table.

Solution 2.

EffortLogger 2.0 has implemented planning poker sessions to run directly off of the same software. This will increase efficiency and lead to less software usage overall.

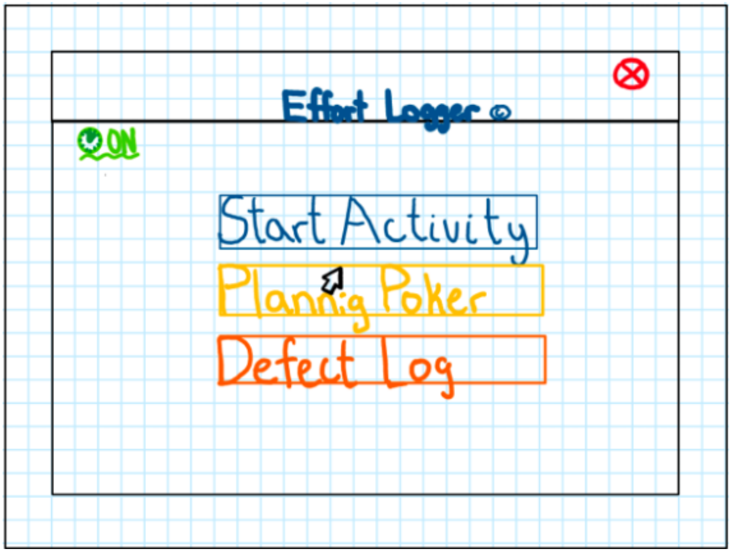
Scenario 3. Security of Important Data

A company discovers that a data leak on their new project information. The old version of the effort logger did not provide enough security measures and the data leakage happened because of it.

Solution 3.

EffortLogger 2.0 has implemented extensive security measures so that the information that is important to the company will stay within the company.

Storyboard
Planning Poker

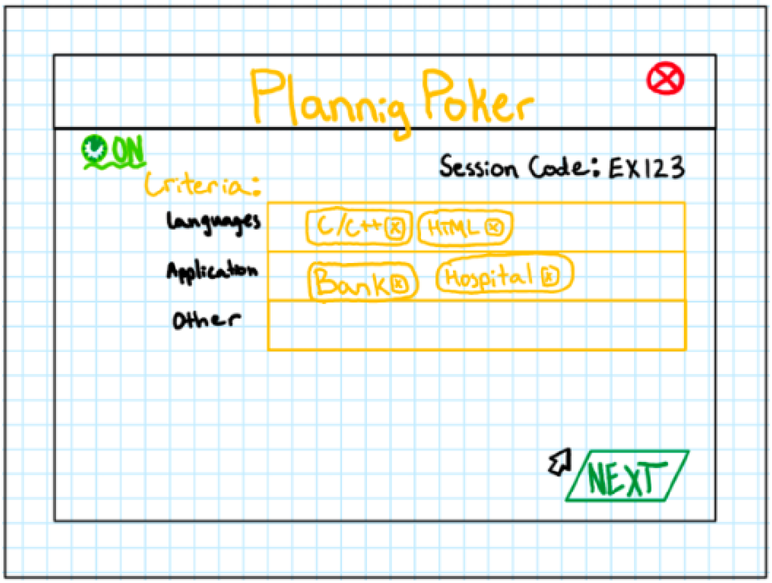


Description: The starting Screen from which each employee lands at the opening of the application This information on the starting screen gives the user multiple options to continue to different functionalities of the application. The clock at the top left can be triggered when any of the buttons, start an activity, planning poker, or defect log

Planning Poker Button redirects to planning poker with a team

Start Activity redirects to an individual log of activity for the project

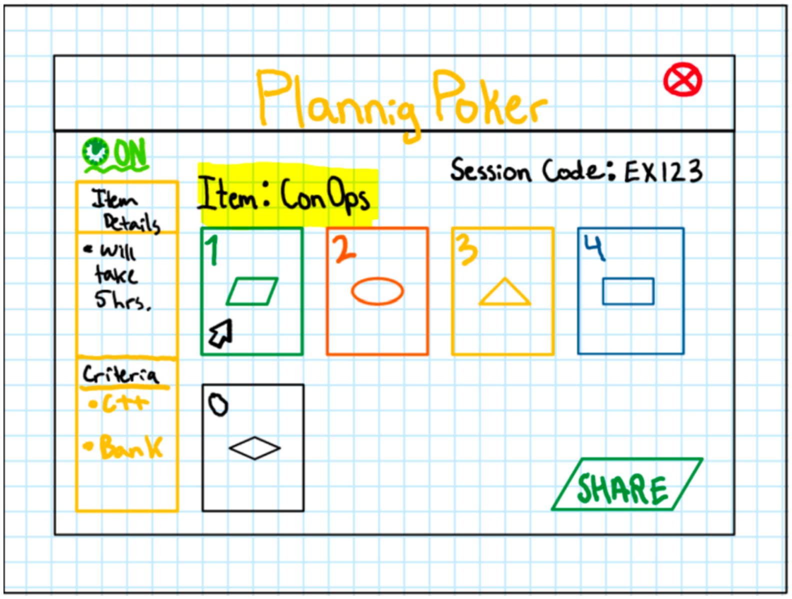
The defect Log redirects to an individual log of defects for the project



Description: The opening page of Planning Poker first displays multiple options for criteria in a project. A team can in real-time connect with the planning session The session code at the top right This code is how teams connect to discuss items on the same project

Criteria section This is a menu of criteria the user can select to determine things already known about the project. This information is winnowed to other sections of the planning poker

Next Button Goes onto the next section of the planning



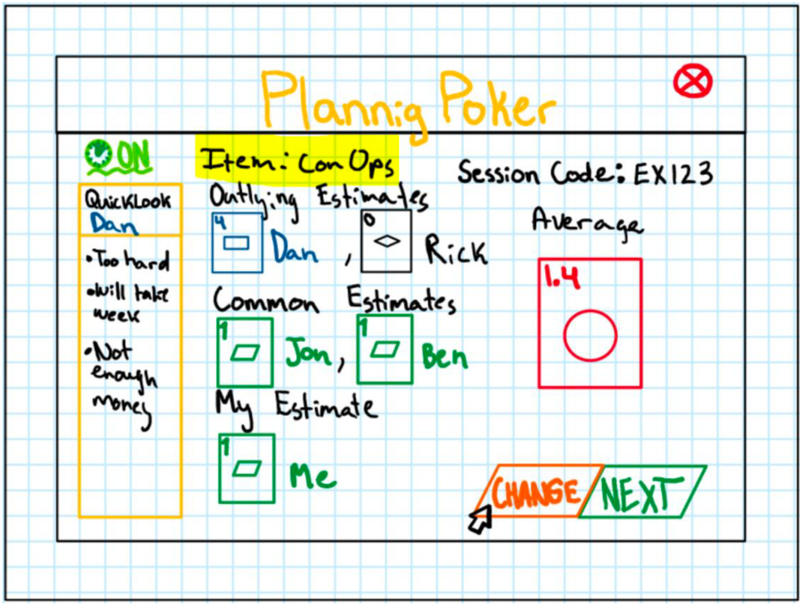
Description: The user plays planning poker by selecting different cards with different weights for an item or feature in a project. Users must write down reasoning and details about items. Know criteria will be displayed for all users in the team group Item Details/Criteria section Item Details is a textbox the user writes in. This information is saved for the user

Criterion is a list of known things about the project

Poker Cards Clickable cards with weights for the item are displayed

The weights of the item are saved for that specific user.

Share Button One card is selected and item details filled in, the user can share their estimate



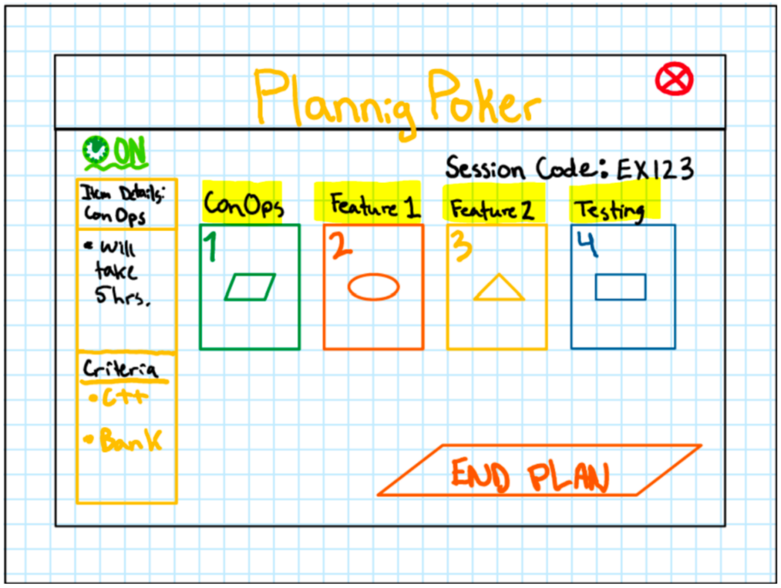
Description: Only when every user in a team group is done with their answers, all team members' cards be displayed under Outlying Estimates, Common Estimates, and My Estimate. QuickLook This text box displays team members’ reasoning for their weighting once a team member card is selected

Team Cards Team members’ weighting is displayed under the appropriate section and once selected display information on the QuickLook text box.

Average Card Displays unique card of team members’ average weighting

Change Button Once members discuss reasoning over a third-party app like Slack or in person, members can change weight to reach unanimity by going back to the previous page

Next Button Once unanimity has been reached, all team members can select to go to the next item and iterate the previous two steps for each item in the project.



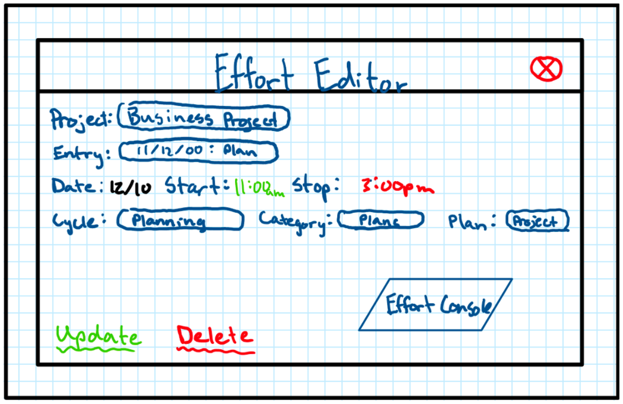
Description: The final page for planning poker displays the final information about all items in the project Item Details This box displays item descriptions specified by all team members for each item once the item card is selected

This information is saved for each feature but is not team member-specific

Criteria Displays known things about the project

End Plan Button Exits the planning session and goes back to the main menu of the effort logger.

Effort Log Editor



Project: Business Project

Entry: 11/12/00 : Plan

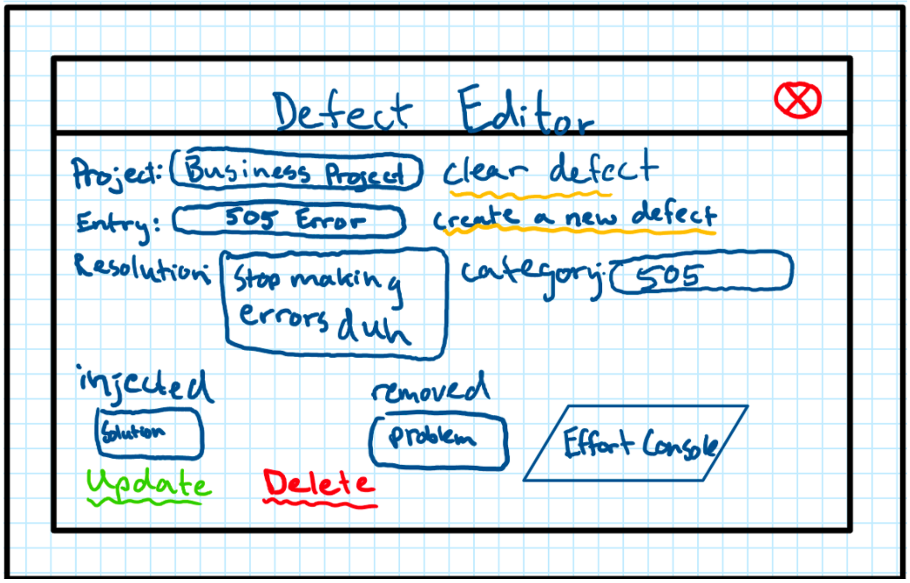
Date: 12/10 Start: 11:00am Stop: 3:00pm

Cycle: Planning Category: Plans Plan: Project

Description: The user plays planning poker by selecting different cards with different weights for an item or feature in a project. Users must write down reasoning and details about items. Know criteria will be displayed for all users in the team group Item Details/Criteria section Item Details is a textbox the user writes in. This information is saved for the user

Criterion is a list of known things about the project

Defect Editor



Project: Business Project

Entry: 505 Error

Resolution: Stop making errors duh

category: 505

injected: Solution

removed: problem

Description: Defect Editor edits and adds defects that come across during the user's activity logs
Project/Entry Specify specific entries to be edited or created in a project

Resolution Text Box Write possible solutions to defect

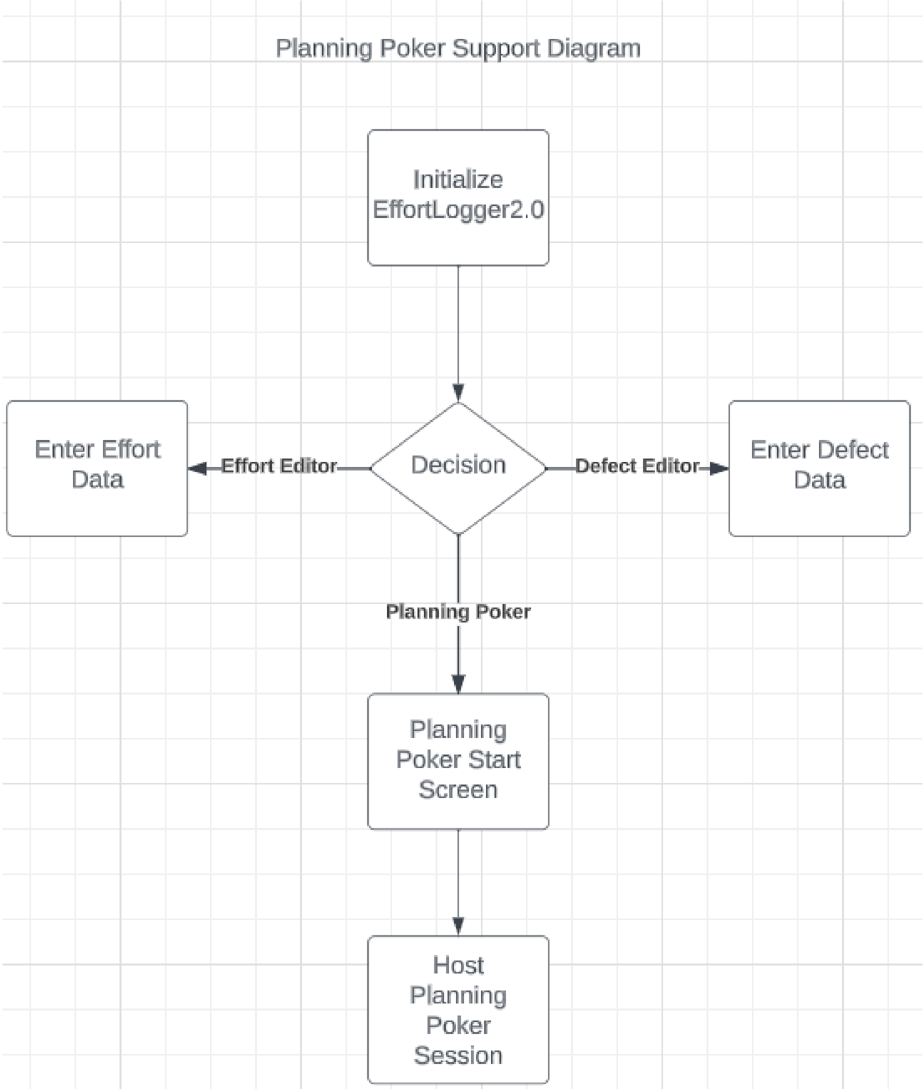
Category The type of error is selected or edited

Inject/Removed Shows where the error was encountered and where the solution will be encountered

Update/Delete Update error log

Effort Console Redirect to the effort logger menu

2.3. Activity Diagram


Planning Poker Support Diagram
Initialize EffortLogger2.0
Decision
Effort Editor
Enter Effort Data
Defect Editor
Enter Defect Data
Planning Poker
Planning Poker Start Screen
Host Planning Poker Session

3. Requirements

3.1. User Stories

User Story 1.

Title: Protecting Employee Privacy while Maintaining Transparency

Story: As a member of the process improvement team, I want to ensure that EffortLogger2.0 maintains a balance between individual privacy and transparency in data reporting so that we can continue data-driven decision-making and improvement efforts effectively.

Acceptance Criteria:

1. The system must anonymize all individual effort and defect reports before they are accessed by any team, project, program, or organizational analysis tool.
2. Identifying information about individuals (e.g., developer 1, engineer 3, supervisor 2) should be included in individual effort and defect reports for targeted improvement efforts and knowledge sharing.
3. When there are insufficient reports to ensure anonymity, access to the source data must not be provided, and summary data should not reveal individual identities.
4. The process flow of data from individuals to analysis tools must be transparent, and a clear explanation of how privacy is maintained should be provided in user-friendly terms.
5. An audit trail should be maintained to track access to individual-identifying information, ensuring accountability and compliance with privacy standards.

User Story 2.

Title: Facilitating Targeted Improvement Efforts

Story: As a first-level supervisor, I want to ensure that EffortLogger V2.0 provides the capability to direct improvement efforts effectively by including identifying information in individual effort and defect reports so that we can enhance predictability and gather best-practice insights from high-performing groups.

Acceptance Criteria:

1. The system should allow the inclusion of identifying information about individuals (e.g., developer 1, engineer 3, supervisor 2) in individual effort and defect reports.
2. Reports should be accessible to authorized users and improvement teams to

analyze performance data.

3. Effective improvement efforts should be directed toward groups where predictability improvement is needed based on the analysis of individual identifying data.

4. High-performing groups should be recognized, and their best practices should be documented for knowledge sharing.

5. The system should support data-driven decision-making by providing tools for analyzing and visualizing performance data

User Story 3.

Title: Improving Planning Poker Sessions

Story: As a member of our scrum team, I want to ensure that EffortLogger V2.0 improves the poker planning sessions so that our scrum team can effectively and quickly specify the criteria of the project and the weights for each item.

Acceptance Criteria:

1. The system should allow users to input project criteria such as programming language and application domain before a Planning Poker session.
2. The tool should filter historical data based on the specified project criteria, presenting only relevant items for estimation.
3. During the Planning Poker session, the system should enable users to quickly assign weights (e.g., 0 through 4) to each item, indicating its relevance to the current user story.
4. The application should have a stylized look and should share the teams poker card weight once it is time to share.
5. A weighted average calculation should be performed automatically, reflecting the contribution of each item to the story points.
6. Users should be able to adjust weights during the session as discussions progress.
7. The two-step process should significantly reduce the time spent on accessing personal repositories and manual calculations, making Planning Poker more efficient.

3.2. Operational Requirements

- Operational Requirements: Document the requirements for more technical people using the concepts from the textbook (functional requirements?)
- The system shall have a list of projects from which the user can select from.

- A user shall be able to start and stop their activity and that activity will be saved to a project.

- Each employee using the system shall be uniquely identified by their employee ID

- Effort and Defect logs contain information that identifies their corresponding project and employee

- The system's authentication and access control features are designed to ensure a secure user experience.

Logging and Tracking: Access control is implemented with a focus on role-based permissions tailored to different user categories. In terms of logging and tracking, users can effortlessly log their efforts with detailed date and time stamps. The System employs a version control system to meticulously track milestones and tasks, providing a comprehensive audit trail.

Planning Poker: The Planning Poker functionality is geared towards real-time collaboration, supporting both synchronous and asynchronous participation. To facilitate seamless communication during planning poker sessions, the system integrates with collaboration tools. For program managers, the system offers robust reporting and analytics capabilities. Customizable reports enable in-depth analysis of effort distribution and overall progress.

3.3. Quality Requirements

The essential characteristics that have been carefully considered and determined are included in the Quality Requirements. As the amount of data and the number of concurrent users increase, a high priority has been given to guaranteeing optimal speed and scalability. Maintainability is emphasized in the architecture, which features a modular and well-structured codebase that makes upgrades and future improvements easier to manage and beneficial to administrators and developers in the future. The user experience is improved by prioritizing usability and accessibility, which ensures an easy-to-use interface and adherence to accessibility guidelines. Regarding integration and teamwork tools, the program is made to work well with poker session scheduling, encouraging and optimizing user ease with maintenance. Sturdy performance and monitoring logging systems have been put in place to thoroughly record pertinent events and mistakes, which is essential for auditing. In order to maintain perfect data consistency, the program sets a high value on data security, guaranteeing database dependability and transaction integrity, especially during crucial activities like effort entry insertion and deletion.

4. Architecture

4.1. Architectural Overview

Logical View:

- Each screen is equipped with a dedicated controller class. Effort and defect entry data are retrieved from the database and stored as distinct classes. SQL statements are generated inside functions based on various parameters.

Process View:

- The system adopts an event-driven architecture, responsive to user input for executing actions. It establishes communication with the MySQL database. The system takes in user input, and creates SQL statements that are sent to the database. There are also checks in place to handle improper inputs.

Development View:

- Individual screens exhibit a high degree of independence. The system comprises three primary components: SQL database definitions, controller classes, and the graphical user interface (GUI). Notably, the GUI and SQL data operate independently of each other.

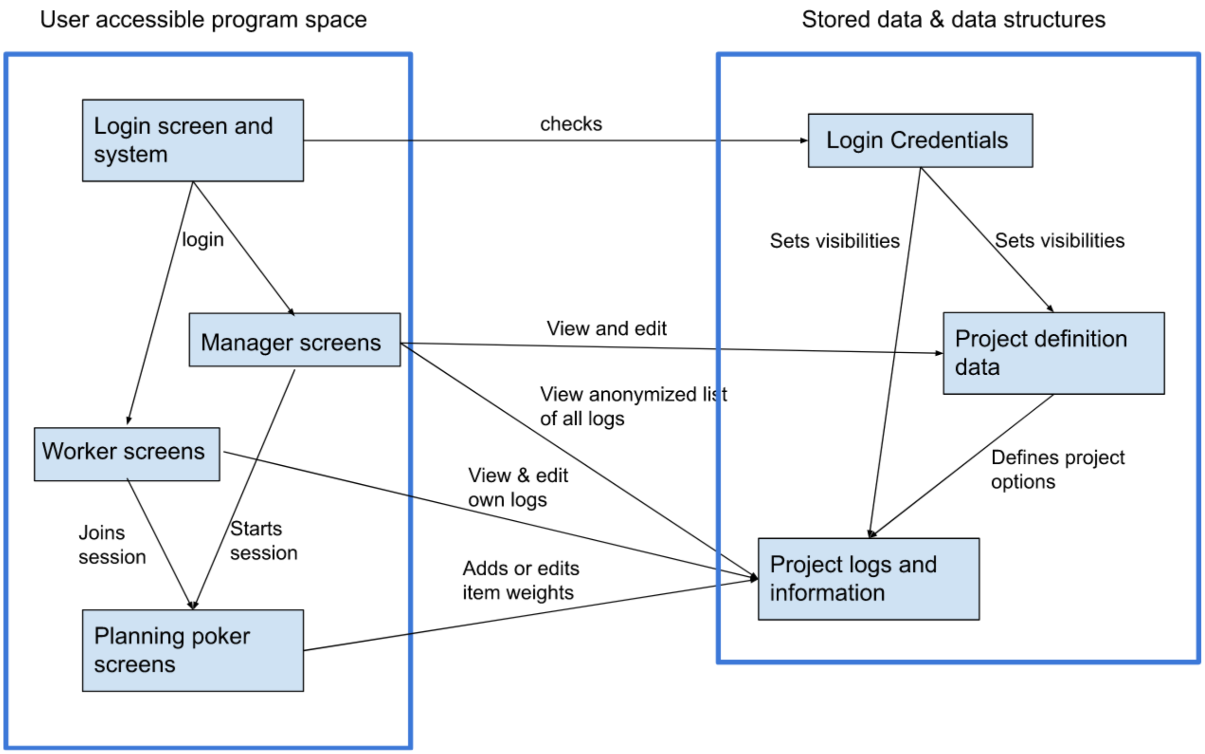
Physical View:

- The program is coded in Java, abstracting away specific hardware details to enhance compatibility across diverse systems. However, this compatibility comes at the cost of requiring Java installation on the host computer. Additionally, due to Java's limitations, the program cannot perform multi-core operations, resulting in potential slowdowns for sufficiently large databases.

4.2. Architectural Elements and Rationale

The architecture of this project was formatted from top-down using the object-oriented architecture. The top was the user/client and the bottom was the information stored in the database. The middle layers were the different applications that the client would manipulate that created different objects of a certain class that would later be displayed in the form of a table or in a combobox in a fxml file. That object information would then be stored in a SQL database.

Architecture



The foundation of the effortLogger v2 system lies in its login screen, a pivotal architectural element designed to regulate access and user identification. This screen, featuring text boxes for the user's name and password, serves as a gateway by authenticating credentials against stored data. The rationale behind this structure is to ensure data protection, allowing only authorized users to interact with the database. This identity-checking mechanism forms a crucial layer of security, controlling the flow of users into the system.

The worker screens, encompassing the effort console, effort log editor, and defect console, represent a thoughtful architectural decision aimed at providing a smooth transition for users accustomed to effortLogger v1. These screens maintain the familiar functionalities of the previous version, allowing users to create, check, and edit their logs with minimal disruption. The rationale behind this approach is to optimize user experience, minimizing the learning curve for workers and facilitating their engagement with the system.

Manager screens extend the architectural design by incorporating two additional interfaces, namely an anonymized list of all project logs and a platform for accessing and modifying project definitions. This architectural expansion serves a strategic purpose – enabling effortLogger v2 to seamlessly integrate planning poker sessions. By incorporating these screens, the system caters to the managerial aspects of project oversight and planning, enhancing the overall functionality for project managers.

The planning poker screens themselves emerge as a distinctive architectural element, serving the specific purpose of facilitating the scrum team's planning poker sessions. The manager-centric control over session initiation and the ability for workers to participate in voting on deliverable weights highlight the deliberate design to emulate face-to-face planning poker sessions. The rationale behind this architectural choice is to replicate a familiar and effective decision-making process within the digital framework.

The architectural underpinning of login credentials emphasizes secure data storage without a viewable representation. This design choice aligns with the purpose of authentication, checking whether the entered name and password combination matches stored credentials. The rationale is clear – to uphold user privacy and authenticate individuals before granting access to sensitive data.

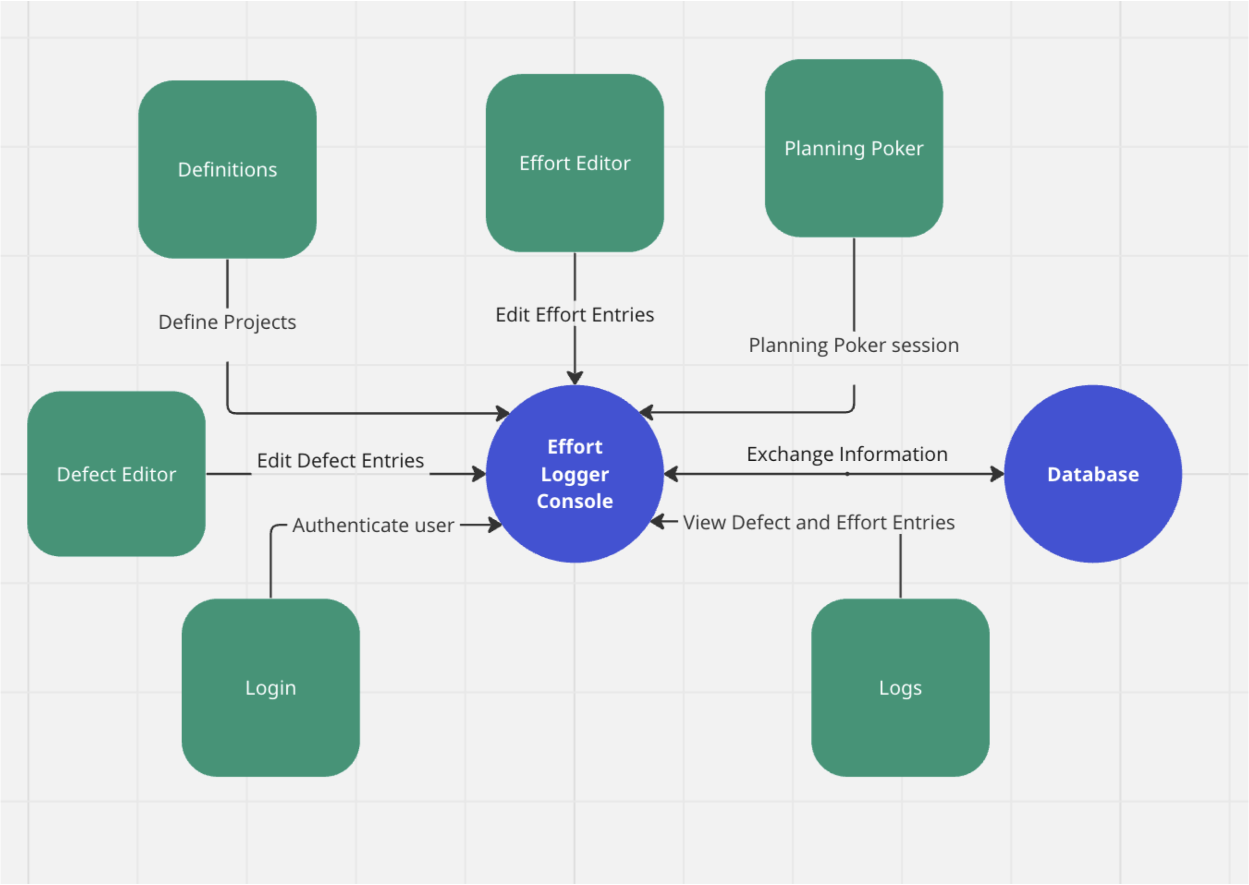
The architectural approach to project definition data restricts editing privileges to the project manager, emphasizing a controlled and hierarchical access structure. This deliberate restriction aligns with the rationale that workers, focused on project tasks, should not alter project parameters, promoting effective project management and maintaining data integrity.

Finally, the architecture of project logs and information introduces a viewability ID as a key element in data accessibility. This design decision ensures that access is contingent on validation against the currently logged-in user. The rationale behind this architectural choice is to enforce controlled access, promoting security and safeguarding project data integrity within the effortLogger v2 system.

5. Detailed Design

5.1. System Context and Interactions

Context Diagram:



Descriptions

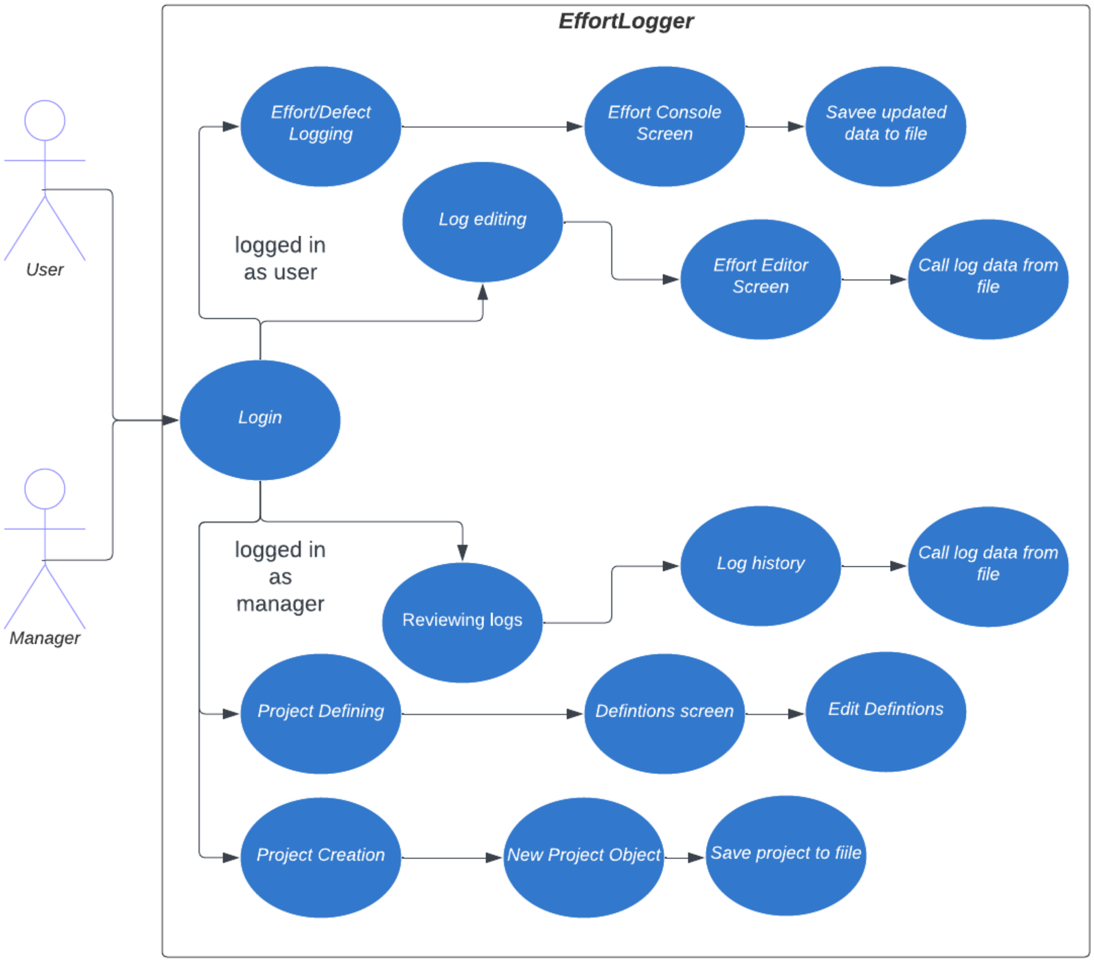
- **Login Screen:** The architectural foundation of the effortLogger v2 system is established with the login screen, which regulates user access and identification. The detailed design elements of this screen include text boxes for user name and password, emphasizing a secure authentication process against stored data. The detailed design seamlessly flows from the architectural decision to prioritize data protection and ensure that only authorized users can interact with the database. The choice to focus on secure identity checking becomes a guiding principle, influencing the design of the login screen to fulfill this crucial role in the system.
- **Team Member Screen:** The detailed design elements of these screens prioritize user experience by retaining familiar functionalities. The flow from architecture to detailed design ensures minimal disruption for workers, allowing them to create, check, and edit logs with ease. This user-centric approach in the architectural decision continues into the detailed design, optimizing the experience for workers and facilitating engagement with the system.

- **Manager Screen:** The manager screens extend the architectural design to incorporate additional interfaces for project logs and project definitions. This expansion strategically enables the seamless integration of planning poker sessions. The detailed design elements of these screens cater to managerial aspects, facilitating project oversight and planning. The flow from architectural decisions to detailed design aligns with the strategic purpose of enhancing functionality for project managers, showcasing a cohesive approach to system design.
- **Planning Poker Screens:** The planning poker screens emerge as a distinctive architectural element tailored for facilitating scrum team planning sessions. The detailed design elements reflect this purpose, with manager-centric control over session initiation and worker participation in voting. The flow from architecture to detailed design ensures that the system replicates a face-to-face planning poker experience digitally, aligning with the rationale to emulate an effective decision-making process within the digital framework.
- **Data Storage and Access Control**: Architectural decisions regarding secure data storage and access control influence detailed design elements in various components, For example, in the design of login credentials, the architecture emphasizes secure data storage without a viewable representation, aligning with the detailed design elements that prioritize authentication and user privacy. In Project definition data, restricting editing privileges to the project manager guides detailed design decisions, enforcing a controlled and hierarchical access structure for effective project management.

Interactions

- **Team Members:** The developers of the system would engage with the GUI design elements created using the FXML files and work on the implementation of user interface elements, ensuring that the design aligns with the usability principles. The team members interact with the user interface to log their daily efforts. They utilize the elements to enter and manage their contributions as well as provide a real-time record of their work.
- **Project Managers:** Project managers interact with the manager screens designed for project oversight and planning. They utilize the interfaces for accessing and modifying project definitions and reviewing anonymized project logs. The detailed design elements cater to their managerial roles, providing tools for effective decision-making and project management.
- **Developers:** Developers implement the architectural decisions. They translate the high-level design into the login screen, worker screens/manager screens and Planning Poker/Effortlogger screens according to the specified design principles.

5.2. Use Cases



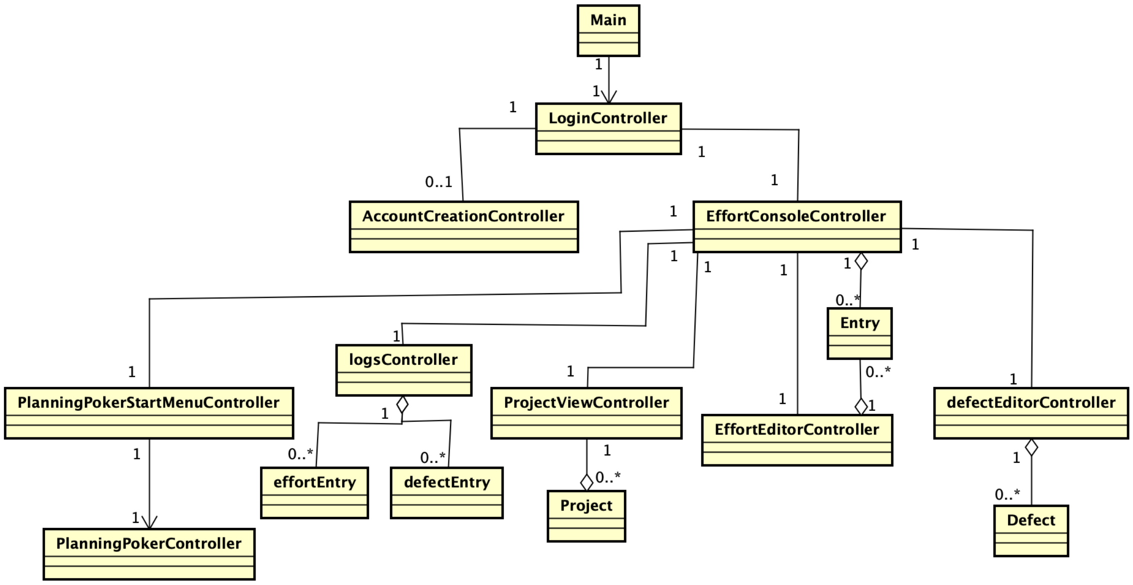
EffortLogger V2 represents an advanced project management and performance tracking tool with a primary focus on ensuring employee privacy, robust data security, and agile development practices. This upgraded system prioritizes user-controlled data sharing and employee anonymization, providing a secure environment for efficient project management and facilitating data-driven decision-making processes.

The functionality of EffortLogger V2 adapts dynamically based on the user's role, whether they are a regular user or a manager. For managers, the program enables the review of past project logs and empowers them to create and define projects. In contrast, regular users can engage in activities such as creating and editing effort and defect logs. This role-based distinction ensures a tailored experience that aligns with the specific responsibilities and permissions associated with each user type.

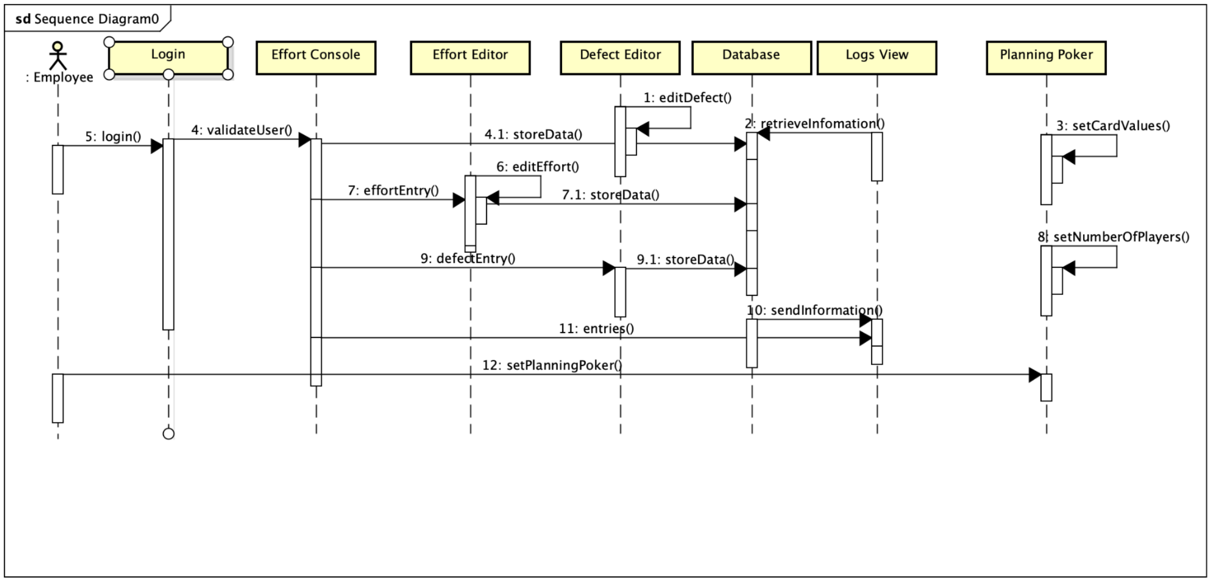
Project creation within EffortLogger V2 initiates the instantiation of a Project class, followed by the saving of this project instance to a file. The process of reviewing past logs involves loading the relevant project from a file and displaying it on the logs screen for comprehensive analysis. Project details, crucial for effective management, can be defined in the definitions screen. This action not only loads the project details but also ensures their secure storage by saving them to and retrieving them from a file.

Effort and defect logs are integral components of the system, and their creation involves utilizing dedicated screens for each. These logs are then saved to their respective projects, contributing to a well-organized and comprehensive project management framework. Editing effort and defect logs follows a systematic process wherein the log information is loaded into the editor screens. Once necessary changes are made, the updated logs are saved back to the project, ensuring that the modifications seamlessly integrate into the overall project data structure. This approach emphasizes a user-friendly and efficient workflow, enhancing the overall user experience within EffortLogger V2.

5.3. Class Diagram



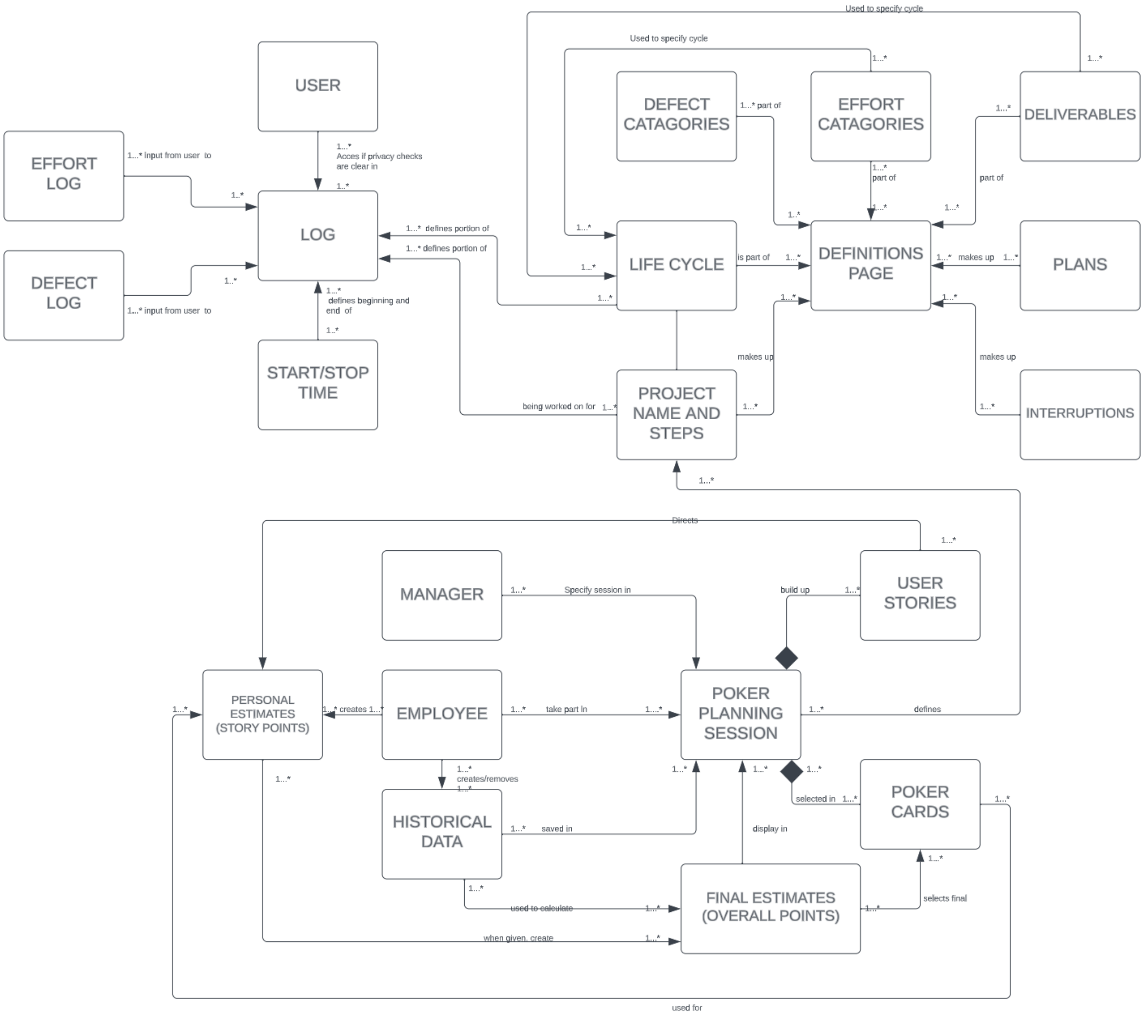
Sequence Diagram:



5.4. Supporting UML Models and Diagrams

Team Project Phase 6

Detailed Design



6. Implementation

6.1. Structure and Naming

The graphical user interface (GUI) for the application was meticulously crafted utilizing Scene Builder, accompanied by the generation of corresponding FXML files to meticulously align with the design specifications. Each distinct scene or screen within the application is intricately associated with its dedicated controller, manifested as a Java class. These controllers proficiently oversee and handle all events precipitated by user inputs, ensuring a streamlined and responsive user experience.

Furthermore, an extensive set of methods was developed to effectively manage the interactions between the system and the SQL database. Notably, for operations such as the insertion and deletion of effort entries, meticulously designed methods were implemented to construct the requisite SQL injection statements, emphasizing precision and security in handling database transactions. This approach ensures the integrity of data operations and establishes a robust foundation for seamless interactions between the application and the underlying SQL database.

6.2. Verification, Validation, and Testing

Beginning with unit testing, each unit is tested independently of the rest of the application, ensuring that the focus is solely on its functionality. Followed by Integration testing in which previous prototypes went through interaction testing, where collaboration and interactions were evaluated between different modules and components; data flow testing, where the flow of data between integrated components is correctly passed and processed; testing interfaces, where the UI between components are functioning correctly between the main, .controller, and .fxml files. During integration testing, incremental testing was followed in which various components (UI, database, workflow, etc.) were integrated and tested incrementally until all prototypes were combined.

Towards the end of the development process, all integrated components were ensured to provide a seamless and satisfactory user experience through various end-to-end testing techniques. Scenario-based testing was implemented in which test cases were designed to represent specific scenarios or user journeys through the application, covering a range of functionalities and potential use cases. This method of user experience testing was performed by manually testing user interfaces to ensure that the interfaces are responsive, functional, and seamless throughout application navigation. Finally, the program underwent testing under multiple environments for consistency and overall code quality.

6.3. GitHub Repository and Contents

Implementation

Link to the GitHub Repository: https://github.com/ZacharyLitwin/CSE360_Tu37

7. Demonstration

7.1. Overview

-Team Screencast link:

https://drive.google.com/file/d/1ONF4qJ2GWLSCXGNlBtnxUzhnRbg9xeCV/view?usp=sharing

7.2. Key Features

Screen Cast 1- Alma Babbitt:

https://drive.google.com/file/d/1lbFBbBhfPYOyKqN31jSh2rD01M_UraCw/view?usp=sharing

Screen Cast 2- Ishan Yelnoorkar:
https://drive.google.com/file/d/1vD43N7BUOJL-AutwzB1Y55_IeVxT0KvQ/view?usp=sharing
Screen Cast 3- Karryl Dumalag
https://drive.google.com/file/d/1LvnzIwGs44Y-0GDcynfGttjxoP5HMaL0/view?usp=sharing
Screen Cast 4- Trevor Huss:
https://drive.google.com/file/d/1FwnXDsjlKgrzJJ4KrmUKMpUsz8p-4oxw/view?usp=sharing
Screen Cast 5- Zachary Litwin:
https://drive.google.com/file/d/18K6eQd4UrXB3u0ihxqmmmzNIdaQpveFu/view?usp=sharing

8. Conclusion

8.1. Overview

In conclusion, the completion of this EffortLogger2.0 Project will grant access to more efficient ways of tracking effort and defect data for companies in a much more streamlined fashion. The final prototype has been completed and the proposed problems have been solved. EffortLogger 2.0 will surely improve the efficiency of the employees who use it. The added functionality of using SQL databases and Planning Poker software will make the lives of the developers a lot easier and will make performing tasks with data much less stressful. The power of EffortLogger 2.0 will hold strong for large corporations and will make for a long-lasting product. Companies adopting this software will come to appreciate the investment they made in their companies future.

8.2. Lessons Learned

Some of the lessons we have learned are very noteworthy and should be included in this document. The strengths of an SQL database encompass several key aspects. Firstly, the compatibility between Java and SQL enhances the database's versatility, allowing seamless integration and interaction between the two technologies. Additionally, the ease of creating a user interface is facilitated through tools like Scene Builder, simplifying the development process and enhancing the overall user experience. Security of data is of paramount importance, and employing SQL databases provides robust mechanisms to ensure the confidentiality and integrity of sensitive information. The optimization process of a database program is a critical consideration, involving the fine-tuning of SQL queries and database structures to enhance performance. Moreover, recognizing the significance of planning for scalability is crucial, ensuring that the SQL database can efficiently handle growing volumes of data and increasing demands over time. Collectively, these strengths contribute to a resilient and efficient SQL database system.

8.3. Recommendations for Improvement

One particularly noteworthy and revolutionary development is the use of artificial intelligence-powered predictive analytics. This improvement would use previous data to successfully assist decision-making by giving users predicted insights about project timeframes, the best way to allocate resources, and possible bottlenecks. Furthermore, making the application more accessible via a mobile platform will enable users to conveniently engage in poker session preparation while on the road, improving responsiveness and flexibility. It would be possible to further optimize workflow operations and foster compatibility and interoperability with a variety of project management tools by expanding integration support to encompass a larger range of management applications. Moreover, it is imperative to design a user onboarding process that is straightforward and provides new users with complete instructions on the functionalities and user interface of the program. This will guarantee a seamless and effective learning curve. Last but not least, bolstering security protocols with sophisticated authentication techniques will strengthen user authentication and create more security layers to secure user accounts and confidential information. Together, these suggested enhancements seek to improve the application's usability, security posture, and functionality while bringing it into line with the changing demands and preferences of its user base.

-

9. Appendix A: Credit Sheet

Team Member Name	Contributions
Alma Babbitt	Login Data and UI EffortConsole Beginnings Definitions Page Project Creation Ability Use Case and Architecture Individual Screencast 1
Ishan Yelnoorkar	Defect Console Logs Screen Sequence Diagram, Class diagram Individual Screencast 2
Karryl Dumalag	Implementation section Account / Login Controller ^ Individual Screen Cast 3
Trevor Huss	Planning Poker Screens Individual Screencast 4 Problem to be addressed Proposed solutions
Zachary Litwin	EffortLogger Console screen Effort Editor screens Team Screencast Individual Screencast 5

10. Appendix A: Current Team Norms

Unsigned Norm Agreement:

https://drive.google.com/file/d/1u-mFyAV-puvFaXHqHHKCrbE5Cu1RqBAJ/view?usp=sharing

Goals

- The team will try to abide by the client's requirements and put forward its best efforts to make that a reality

Meeting and communication norms

- Class time will be utilized to gather notes and information regarding software project management processes.
  - Class time will be utilized to brainstorm how these ideas could be used in the project
  - The team will meet every Friday to perform a scrum, discuss, manage backlog, and plan future action
    - The team will meet at Noble Library, per convenience and the meeting will last 2 hours
- Apart from weekly meetings, the team will communicate via discord/text chain, giving minor updates every time, a task is complete
  - The team will communicate effectively and swiftly to avoid delays in work
  - During holidays and long weekends, work will be allocated per the team member's unavailability for the holiday. Work will be redistributed and overworked individuals will be compensated with fewer workloads for the following week after the holiday

Work norms

- The team will work 5 hours every week to ensure timely delivery of deliverables
- The team will split work according to the necessary skills required for the task to be completed
- In case a member of the team is not getting work done, they will be given a warning first, and then a mail to the TAs will be sent for a repeat offense
  - The team will set deadlines based on the urgency and time requirements of a task
  - Every week a different member of the team will be allocated to proofread work that has been done so far
- Everyone is allowed to work in their manner as long as progress is made, and it does not impede the group's ability to make progress
  - All team members are expected to adhere to team norms and meet its expectations

Decision Making

- For a decision to be made, the majority of team members should agree with the agenda put forward, and simultaneously try to understand and help understand why the other's point of view may or may not work with the task at hand
- Team members will listen actively and take into consideration everyone's point of view, as well as try to resolve disagreements

Appendix B: Current Team Norms

We, group members of Tu37 have agreed to follow the terms listed above and plan to adhere to them until the culmination of this project. Our listed names below indicate our acceptance of the norms and are used as our digital signature.

Member 1: Alma Babbitt

Member 2: Zachary Litwin

Member 3: Trevor Huss

Member 4: Karryl Dumalag

Member 5: Ishan Yelnoorkar