Milestone 1 – Project Proposal

Team Name:
Plaintext Letters … and Nothing But

Team Members:
Matt Davenport
John Keller
Sam Mauck
Luke McDonald

Description:

The team intends to build a secure, self-destructive text-based communications platform. This service will allow users to anonymously communicate with each other for an agreed upon period of time, after which the conversation will be irreversibly deleted. This concept has become popularized over the past few years, but we believe having both users choose the duration is the best of both worlds. If the content in the message is sensitive, they can elect to choose a shorter period of time. The communication will be secured using end to end encryption such that only the individual users will be able to read the messages; the messages pass through the server encrypted.

In other communication applications, there are often far too many unused features. Who wants to add a silly sticker to a message? We don't believe that this frivolous feature is needed, and plan to build an application which is limited to plain text. This means no silly emojis, no pictures of cats, and no songs, which only serve to distract the users from their conversations. We are bringing back the basic and retro enjoyment that many people grew up using, while layering in a level of security which is sure to protect your conversation.

Vision Statement:

To fill the desire for simplicity and security in users who have grown weary of convoluted modern communication software by providing a fast, simple, and secure solution.

Motivation:

In today's interconnected and over-sensitized world, it is hard to come across a communications platform for power users. Almost all of the communications platforms today retain some sort of chat history and have no security, or proprietary security at best. This project aims to be a no-frills, efficient chat app for power users that provides them with security and the reassurance that their private conversations won't ever be sitting on someone else's server.

Risks:

1. One of the biggest risks for an application of this sort is implementing proper security. This can potentially become a huge time sink because cryptography and security is often very dense and abstract but at the same time it is one of the selling points of the application.
2. Another risk for this project is the size of the team. One of the initial group members transferred to a different recitation section, leaving the size of the group at 4.
3. Finally, one smaller risk is not understanding each other's strengths and weaknesses given the fact that the team has never worked together before.

Risk Mitigation Plan:

1. The first, and biggest risk for this project, is security. In order to mitigate this risk, the team will start small and build big as an afterthought. This means that the team will implement basic security features such as login and chat deletion after a specific time period (which is one of the main selling points) and then transition to end-to-end encryption if and only if everything else is done, still resulting in a successful project. In addition, given that this project is more of a proof of concept, the cryptography does not need to be so robust that the project becomes more about cryptography than the chat platform itself.
2. The next risk for the project is the team size. Most teams for this project have five or six members. However, given that this team has only four members, in order to succeed in this project the team will carefully evaluate the amount of work the project will take, and will constantly assess the remaining work and make weekly plans accordingly.
3. The final risk (albeit the smallest one) is a vague understanding of each team member's strengths and weaknesses. This will be resolved quickly during the planning stage of the project such that each team member works with technologies they are familiar with, while still devling far enough into the unknown to learn something from the experience.

Version Control:

The team will be using the three required GitHub repositories:

- Meeting Minutes: https://github.com/lukemcd1994/planb-teamlogs
- Milestone Submissions: https://github.com/lukemcd1994/planb-milestones
- Project Source: https://github.com/lukemcd1994/planb-codebase

Development Method:

The team will use an Agile development method. Each week, when the team meets, the Scrum Master will start out by leading a discussion regarding what tasks have been completed since the last meeting. New tasks will be added to the Trello board as they arise based on each team members' current individual's responsibilities and their respective ability to complete a given task. During sprint planning, each task will be given a point value representing its difficulty. The goal is that each team member takes on a relatively equal amount of points for the sprint. At the end of each sprint the team will reevaluate the project and create new tasks based the current state of the project as well as any new features that need to be added that were not apparent before. The team will analyze how many points gets completed each sprint such and over time evaluate the maximum amount of points that can be completed each sprint in order to provide better planning for future sprints.

Collaboration Tool:

The team will use a multitude of collaboration tools. For direct communication between the team members, the team will utilize text messaging as well as Discord. In order to collaborate on project documents and milestones the team will utilize Google Docs. Finally, in for sprint planning and tracking the team will Trello.

Proposed Architecture:

The project will have two main components. First, for the front-end, the web app will be based on Vue.js and Bootstrap. This will provide the main front-facing interface that the user will be interacting with. The backend of the system will be comprised of Laravel and PHP7 which will be connected to a relational SQL database (which in this case will most likely be MySQL). The front-end will communicate with back-end mainly by using the Vue Router but will also use AJAX calls and WebSocket sessions. This platform will utilize additional technologies such as WebSockets to provide near instant messaging and the Diffie–Hellman key exchange as the basis for end-to-end encryption. The final product will be hosted on a Linux machine that hosts all of the required software including, but not limited to, the database, the PHP interpreter, and the web-server. The web-server of choice for the final product will be nginx. The web server's connection to each individual client will be secured using HTTPS.