4716 Ellsworth Avenue
Apartment 202
Pittsburgh, PA 15213

October 15, 2014

Mr. Thomas M. Keating
Assistant Teaching Professor
School of Computer Science

Dear Mr. Keating:

Please find attached to this letter our team's proposal for ExplainShell for Chrome. The proposal will outline the details of this Google Chrome extension and demonstrate our plan for its execution, as well as testing and evaluation criteria.

The document contains an abstract, an introduction, a literature review, a plan, a discussion of the project's benefits, our approach, evaluation criteria, and the qualifications of team members.

Please direct any questions or comments about the proposal to jrgallag@andrew.cmu.edu.

Sincerely,

Justin Gallagher

enclosure: paper entitled ExplainShell for Chrome

Justin Gallagher, Ted Li, Jacob Zimmerman, Howard Chen
Group 20
October 15, 2014

15-221
Fall 2014
Section A

Writing Assignment 3: Project Proposal

ExplainShell for Chrome

Justin Gallagher (jgallagher@cmu.edu)
Ted Li (wenxuanl@andrew.cmu.edu)

Abstract

This paper is a proposal for ExplainShell for Chrome, a Google Chrome extension which helps users understand shell commands found online. Instead of blindly running commands found on the internet, the user can learn more thoroughly what they do. This prevents malicious commands from being run, and imparts the user with valuable knowledge. The proposal details our specific goals, approach, and schedule for completing the project.

Contents

1 Introduction

ExplainShell for Chrome will be a Google Chrome extension which aids users in learning and analyzing shell commands found online. When looking for help with Unix based systems, many users search the internet for answers, and use commands found on online forums and Q&A centers without the proper understanding of their function. This can result in malicious or insecure commands being unknowingly run by the user. In addition, running commands this way prevents users from learning how to write such commands themselves.

ExplainShell for Chrome allieviates this problem by showing the user a popup window with a detailed explaination of commands found online. Utilizing information from explainshell.com, another open source project the extension will prevent users from copy and pasting to another page, a barrier which might cause users to run commands without checking their function. This project will provide a useful product and allow us to explore Chrome extension development as well as learn how to interact with other open source projects.

2 Plan

Our goal is to develop a Chrome extension that helps users learn the usage of shell commands found online and avoid malicious shell commands. After the user installs the extension from the Chrome extension store, it will scan any web page that the user visits for shell commands in the background, and allow users to simply hover / click on unfamiliar shell commands to view a detailed explanation of the shell commands in a pop-up window. This extension will not only help users learn and familiarize with the usage of nearly 30,000 shell commands, but will also help users identify potentially destructive or malicious commands before executing them.

We will also create a back-end server to store and provide interesting analytics about trending shell commands, trending referring sites and other statistics. If time allows, we will also attempt to develop a more advanced web page parsing technique to identify shell commands in a web page, possibly using machine learning.

For this project, we have set up a Git repository that allows us to coordinate and keep track of code changes easily. We have also set up a Google Drive to keep track of our progress. We will set up team meetings at the end of each phase by email or texting.

3 Benefits

This project will benefit both our team and end users. For users, the project will provide a convenient way to look up and learn the usage of unfamiliar shell commands found online. Considering that shell commands are often powerful and can sometimes be malicious, the project can also potentially automatically warn users about malicious shell commands before they execute the commands.

We will learn to use and familiarize with a lot of technologies in the process as well, including Git, Chrome extension, Javascript and Python. Through the process, we will also contribute to the ExplainShell open source project.

4 Approach

4.1 Methodology

In this section, we will explain the various technologies that we we will be using to complete the project. Through our project, we will develop a Chrome extension and a back-end server that provides analytics about trending shell commands. We will also contribute to the current code base of the open source ExplainShell project.

For the Chrome extension, we will use Javascript and jQuery to scan and parse the webpage that user visits, to identify potential shell commands within a web page. We will then use AJAX to send requests to the ExplainShell API and receive documentations of identified commands. We will also use HTTP + CSS to layout a pop-up window that displays the documentation.

We will also create a back-end server with Node.js. The back-end server will store the history of shell commands that users look up using our extension, using a SQL database. The data will allow us to provide analytics and visualization about popular or trending searches.

4.2 Project Schedule

Specifically, our project will be separated into the following four phases.

Phase 1
We will first build a simple Chrome extension that allows user to look up the usage of shell command by selecting and highlighting any text in the browser window. Upon highlighting, a button will appear next to the highlighted text region. The user may click the button, which will instantly redirect the user to the corresponding usage documentation at the ExplainShell website in a new browser tab.

Phase 2
We will refine our Chrome extension developed in Phase 1. Instead of asking the user to highlight the shell command that they want to look up, the Chrome extension will now automatically parse and search the web page to identify any shell commands. Upon identifying potential commands, the extension will turn the command into a clickable link, which will redirect the user to the usage documentation at the ExplainShell website.

Phase 3
We will contribute to the code base of the ExplainShell project by adding an HTTP API, since this would also help us further improve our chrome extension. With the API, the Chrome extension may now display a pop-up window in the page, whenever the user hovers / clicks on the identified shell commands, whereas previously we always redirect the user to the documentation web page.

Phase 4
We will develop a back-end server that provides analytics that allow users to view trending searches and referral sites. These analytics will provide insight in which shell commands people take most interest in or are confused about.

Please refer to Fig. 1 on page 4 for a Gantt chart of our project schedule.

5 Evaluation

Our goal is to create a user-friendly Chrome extension and to eventually publish it onto the Chrome extension store. Interested users may download the ExplainShell Chrome extension. On one hand, we will ask our friends and classmates to help evaluate our extension, on its utility and ease of use. On the other hand, we will also request feedbacks from users who downloaded our extension from the Chrome extension store.

6 Qualifications

Overall, our team members have been previously been exposed to relevant technologies. Jacob had 1 year of full-stack web development and interned at Bloomberg L.P. working on software engineering. Justin has 2 years of HTML/CSS/Javascript experience and had a summer internship at Microsoft working on ASP.NET web applications using SQL databases. Ted has experience in iOS and web programming and interned at Bloomberg L.P. working on a project related to natural language processing. Howard has strong background in algorithmic thinking and taught Principles of Programming and Software Development this summer.

There may be specific technologies that we are unfamiliar with, but they should be easy to pick up and would definitely be a good learning experience for us.

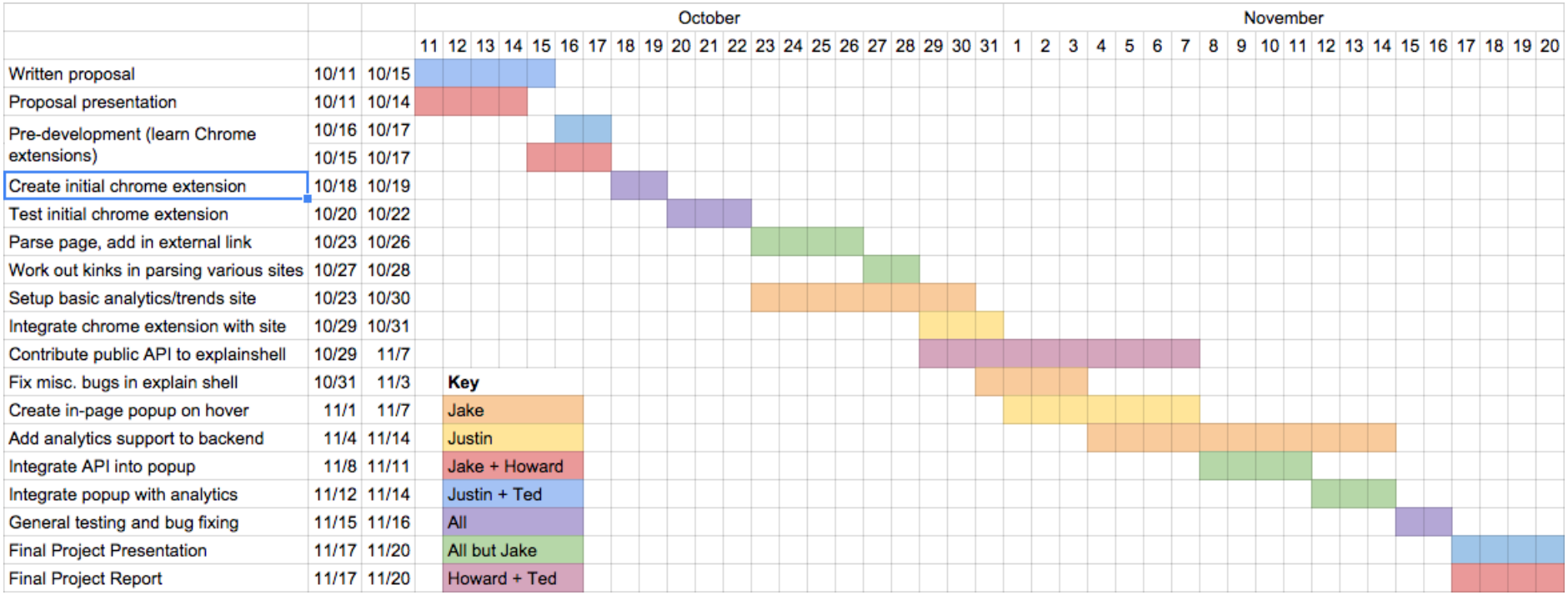

Figure 1: Project time line

References

[1] "ExplainShell project – write down a command-line to see the help text that matches each argument", `http://explainshell.com/`

[2] "Bash One-liners project", `http://bashoneliners.com/`

[3] "Linux commands – beginner's tutorial to the command line environment", `http://linuxcommand.org/`

[4] "TLDP – widespread documentation on all things Linux", `http://tldp.org/`