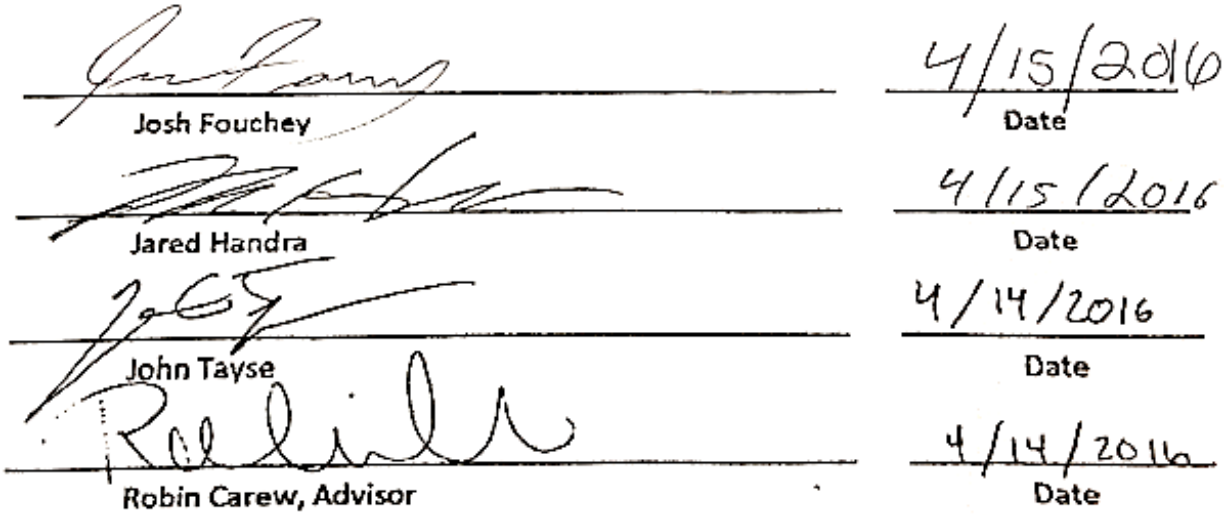
Compiler

by

Josh Fouchey, Jared Handra, and John Tayse

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology



University of Cincinnati
School of Information Technology
College of Education, Criminal Justice, and Human Services

April 2016

Table of Contents

Figures

Tables

Abstract

Compiler seeks to help those within an organization find answers to programming problems. While the Internet can provide answers to coding questions, these answers may not be enough to assist with an internal project. Some question require the experience found within an organization that you just cannot get from the Internet.

Users can post a question and include any source code. Questions use tagging to give insight into the nature of the question like the programming language. Others can post answers and the question poster can select the answer that helped the most. People can also vote up questions and answers they feel are relevant or vote down the things that are not. Users of Compiler receive skill points for interactions and a user's reputation speaks to their legitimacy.

Compiler seeks to connect those with questions to those with answers, all while promoting collaboration and understanding.

Introduction

Developers within organizations come from all backgrounds, and have different levels of experience. We believe we use this to our advantage and create the tools that allow people of all experience levels to help each other on software development related tasks. New developers coming into an organization will have a knowledge base they can consult when they get stuck or when someone more experienced is not available.

Problem

Developers new to a company can struggle when working on projects where proprietary or legacy technology is being modified or upgraded. In a study of IT leaders from more than 3700 companies, respondents said they spend an estimated 72% on support and maintenance of existing infrastructure (Zetlin). Communicating with more knowledgeable developers can sometime be difficult especially when those staff are on different teams or different schedules. In some cases, the people who created the programs departed the company and left no meaningful documentation. This disconnect between those interacting with the program and knowledge of the program only grows as the size of the company becomes bigger.

In educational environments, students with limited programming abilities or those changing focus to software related fields can find homework assignments difficult. Students have access to tutors, but may need a question answered quickly or may not be able to meet with one due to their schedule. Students could also try to get help from a professor, but professors do not have time to meet with every student or may not even have an office on campus. Students can use websites like Stack Overflow or find tutorials, but often students are not presented with enough information to help them solve the assignment.

Solution

Compiler is a web application that fills two main needs, answering users' questions and finding an expert. Compiler allows a user to create a post with the code they are having trouble with, and other users can offer their help. When submitting a new question, the web platform uses a tagging system based on programming language. This allows other users to search based on these tags; so they can offer help in answering questions in their area of expertise.

To solve the second need Compiler uses a matching algorithm that allows the user to be paired with an expert. The user picks specific programming languages or areas of focus they want help in, and are matched with an expert or someone who enjoys helping others in those areas. The goal of this web platform is to create relationships within organizations, allowing more experienced developers to help those with less experience and further expand the sense of community.

The web application is simple and does not burden the users' experience. We did not want the user to feel they had another account they must keep up with and manage. Because of this, we are allowing the user to sign in quickly with Google and Github. If they choose to sign in this way, Compiler pulls information from the user's profiles so that they don't have to enter all their information again. Future features could include the possibility of authenticating with Active Directory or Single Sign-On (SSO) solutions, since our target user base is users within an enterprise.

User Profile

Potential Users:

Technical members within an enterprise organization, specifically but not limited to developers.

Software and Interface Experience:

Compiler has a simple user interface from the user standpoint. The user is required to know very basic knowledge about how to navigate a website. The user also needs to know how to use/read websites like

Yahoo Answers, Quora, etc.

Experience with Similar Applications:

Some potential users may have had experience with similar web applications that involve creating an account and posting/viewing information, especially developers, who have most likely interacted with Stack Overflow.

Task Experience:

The user easily creates an account. They will then have the ability to post their code and receive feedback on it. Users can help other users by reviewing code and providing feedback. The user can also be “matched” to a potential expert to help them with their current or future projects.

Frequency of Use:

Frequency of use is dependent on the needs of the user. A casual user may use the web application to get one question answered and a heavy user may use it to get multiple questions answered.

Key Interface Design Requirements that the Profile Suggests:

- Very quick and simple to use interface.
- Appropriate categorization with “tags” to make it easier to browse questions and accurate search.
- Fluid responsive design to make the application available on many devices.
- An efficient algorithm to match the user with experts efficiently and effectively.

Use Case Diagram

Compiler is a web platform. This requires it to be simple, fast, and convenient for the user to use. We did not want it to become another platform where the user feels like they need to fill out a bunch of information about themselves before they login and start using the platform. The platform is developed

around two major features, as well as a few common features such as signing up for an account and logging in.

Use case number one involves a junior developer asking a question. A senior developer, junior developer, or other expert then provides their solutions. Once the correct answer has been determined, the person who asked the question marks it so other people with the same question can find it anytime.

Use case two involves a junior developer seeking an expert. The user will answer a few questions about what they want help with or what they want to get out of the connection. Senior developers or other experts that want to act as an expert will also answer a similar set of questions. Compiler then takes the information received on either side and match the junior developer with the appropriate expert. A use case diagram for all the cases involved in Compiler can be found in Figure 1 below.

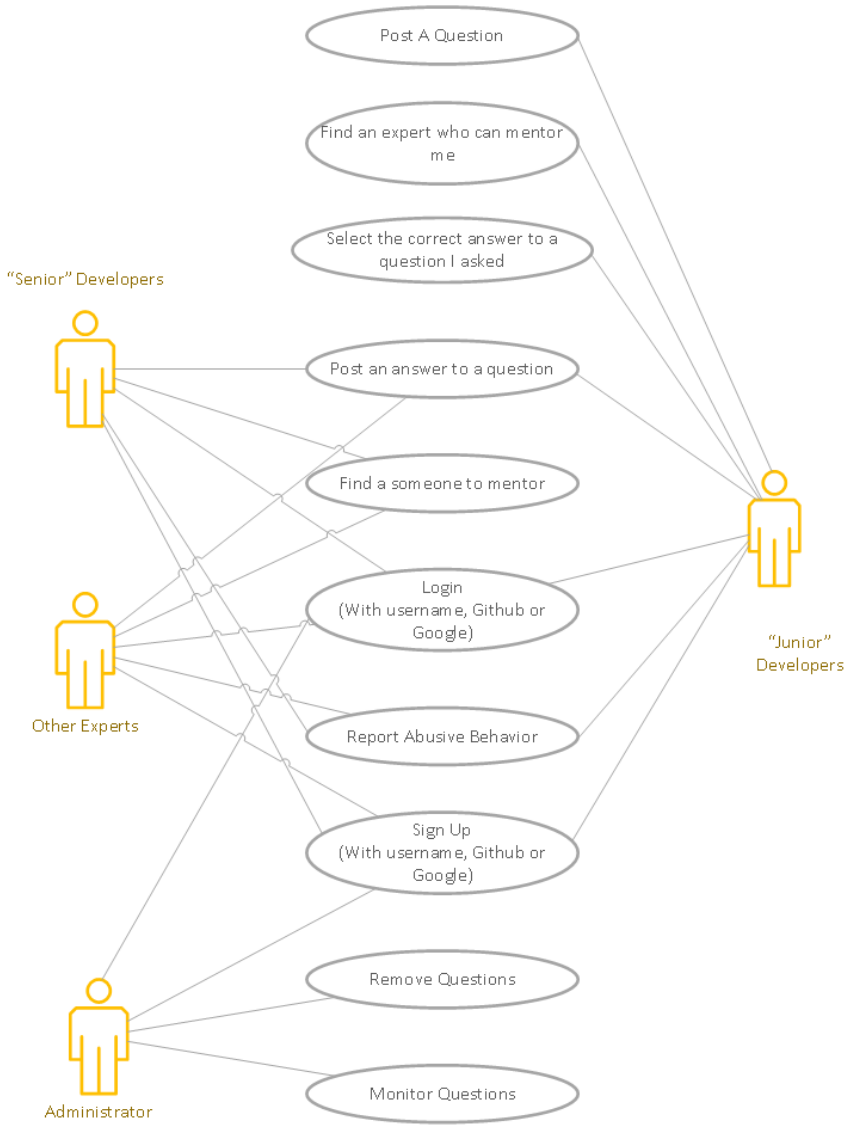


Figure 1: Use Case Diagram

Technical Elements

This application is developed in an open source platform called Meteor. Meteor is unique in the entire technical stack is written in JavaScript ("Introduction | Meteor Guide"). Meteor comes prepackaged with MongoDB, which is a non-relational database and is what we are using to store everything on the backend. On the front end we are using custom CSS styling and JavaScript on top of the Bootstrap framework. Bootstrap provides a great starting place for building the front end of web platforms by providing helper classes for things like forms, buttons, text, etc.

Compiler is for users within an enterprise. Users access the web application in their web browsers and are greeted with a splash screen. The splash screen offers a few details about Compiler, like how it works, whom it's for, and information on specific features. After signing up using Google, Github or creating a username and password, the user logs in and is greeted by a dashboard. The dashboard contains a table with questions asked by other users and currently open. The user can see the tags associated with each question so they can easily find the questions that pertain to them. The user can also search for specific tags, question titles, or question descriptions. If the user cannot find the answer they are looking for, they can ask a question themselves.

When asking a question, the user enters a title, description of the problem they are having, and then add the appropriate tags for their question. A future feature for Compiler could have an admin that makes sure each question is relevant and tagged correctly.

If the user wants more help or would like to meet up with someone in person, they are able to click on the user's username and view more information about the user. They can see when they are available to meet up or if the user is available by email. Basic information like the users rank and what languages they are interested in is also available.

The “Find an Expert” feature allows the user to enter the languages or subject they would like help in and then are matched with someone willing and with experience. The user can then contact or even arrange a meet up with the expert and work together to work through problems together.

Testing

Testing for Compiler is currently done manually, although the possibility for incorporating automated testing exists. The scope of the testing is to test the functionality of Compiler. These tests will be based on the requirements of the project. The objective of the testing is to ensure all functionality is working as expected. For collecting results, all tests will be run by team members and tested for success. The results of tests were aggregated and issues were passed to the appropriate developer(s). If a feature is not working or has not been implemented yet, its test results were ignored. Our test plan brief can be found in Table 1 below with a more detailed test plan and test results found in Table 3 in the Appendix.

User	Feature	Test Description
Authorized User	Account Services	User can login
Authorized User	Account Services	User can logout
Authorized User	Account Services	User can edit account information
Authorized User	Account Services	User can create account with username and password
Authorized User	Account Services	User can login with Github
Authorized User	Account Services	User can login with Google
Authorized User	Account Services	User can edit availability
Authorized User	Account Services	User can edit interests
Authorized User	Account Services	User must be logged in to post a question
Authorized User	Account Services	User must be logged in to comment on a question
Authorized User	Account Services	User must be logged in to vote for an answer
Authorized User	Account Services	User must be logged in to contact another user
Authorized User	Account Services	User must be logged in to view another user's availability
Authorized User	Account Services	User must be logged in to connect to a tutor
Authorized User	Account Services	Poster must be logged in to select answer

Authorized User	Account Services	Poster must be logged in to edit question
Authorized User	Account Services	Commenter must be logged in to edit answer
Authorized User	Combined Functionality	Mentor/Mentee must be logged in to create mentorship
Authorized User	Combined Functionality	User can take Mentor quiz
Authorized User	Combined Functionality	User can enable mentorship feature for their account
Authorized User	Combined Functionality	Visitor can view homepage
Authorized User	Combined Functionality	Visitor can view questions and answers
Authorized User	Combined Functionality	Visitor can search questions
Authorized User	Combined Functionality	Visitor can view user profile
All Users	Site Appearance	Pages are designed consistently
All Users	Site Appearance	All pages have Compiler branding
All Users	Site Appearance	Text is readable on all pages
All Users	Site Appearance	Images have 'alt' attribute

Table 1: Test Plan Brief

Timeline

The initial project timeline for Compiler and based primarily on the winter semester's work can be found in the Gantt chart in Figure 2. The timeline was refined in the spring semester to adjust for the needs to complete the project on schedule and can be seen in the Gantt chart in Figure 3. The team completed deliverables in anticipation of the presentation at the School of IT Tech Expo in April. The timeline includes submitting assignments for the Senior Project Management class, research, development and testing.

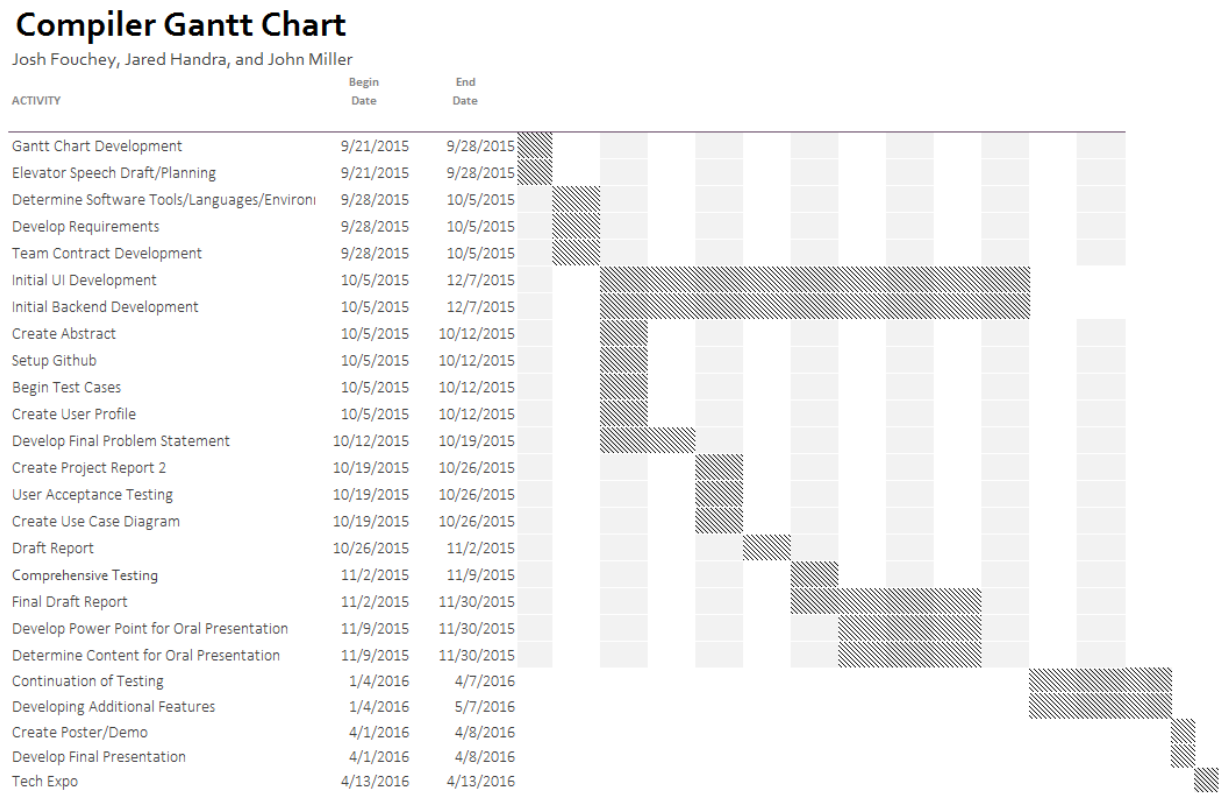


Figure 2: Initial Project Timeline

Figure 3: Revised Project Timeline

Proposed Budget

The proposed budget for this project was less than $100. We had initially hoped to provide a hosted solution for Tech Expo. Instead, we hosted a local version for demonstration purposes. For a private

development firm to attempt this project, we assume the project would be assigned to a team of 5 developers who would work on the project for six to eight months. This equates to 120-180 working days. Each developer would earn $35/hour and works 8-hour days. This amounts to a total cost of wages of $168,000-$252,000. On top of this cost would be the cost of hosting. Again, a private firm would use AWS, which would cost around $50/month. Table 2 below summarizes the budget for Compiler.

Item	Price
Amazon Web Services CloudFront	$50/month
Github	Free
Domain Name	$12.00/year
Meteor Platform	Free
Labor	$252,000
Atom	Free
Total	**$252,612**

Table 2: Proposed Budget

Conclusion

Compiler is an open source application, so that others can continue building the platform. Initially the application was only marketed towards students of the University of Cincinnati but Compiler can be easily expanded to other universities and educational purposes, and enterprise settings. We hope future interested people can carry on the development of Compiler for years to come.

References

"Introduction | Meteor Guide". *Guide.Meteor.Com*. Last modified 2016. Accessed April 6, 2016. http://guide.meteor.com/.

Zetlin, Minda. "How To Balance Maintenance And IT Innovation". *Computerworld*. Last modified 2016. Accessed April 6, 2016. http://www.computerworld.com/article/2486278/it-management/how-to-balance-maintenance-and-it-innovation.html.

Appendix

Comprehensive Test Plan – Below in figure 5 is our comprehensive test plan. It explains exactly the tests ran to ensure the requirements for deliverables were completed.

1	Primary usage limited to authorized users		
	a	User must have a site account, Github, or Google Account	
		i	user can create a site account
		ii	user can log in with a site account
		iii	user can log out from a site account
		iv	user can log in with a Github account
		vi	user can log out from a Github account
		vii	user can log in with a Google account
		viii	user can log out from a Google account
	b	User can manage site account	
		i	user can edit site account email
		ii	user can edit site account password
		iii	user can edit site account display name
		iv	user can recover password for site account
		v	user can delete site account
	c	User must be logged in to edit user profile	
		i	user can edit interests
		ii	user can edit personal information
	d	User must be logged in to interact with questions and comments	
		i	user can post a question
		ii	user can answer a question
		iii	user can vote for a question
		iv	user can vote for an answer
		vi	poster can edit a question
		vii	commentor can edit an answer
		viii	poster can select a best answer
		ix	poster can change best answer

		x	poster can remove question
2	Secondary usage for visitors and combined functionality		
	a	Visitor can view homepage	
	b	Visitor can view questions and answers	
	c	Visitor can search questions	
	d	Visitor can view user profile	
3	Site appearance		
	a	Pages are designed consistently	
	b	All pages have Compiler branding	
	c	Text is readable on all pages	
	d	Images have 'alt' attribute	

Table 3: Comprehensive Test Plan

Technical Solution Diagram – Below in figure 4 is our technical solution diagram. It explains how our application works in technical detail. Because we are using Meteor, all of this works out of the box and fits our needs perfectly.

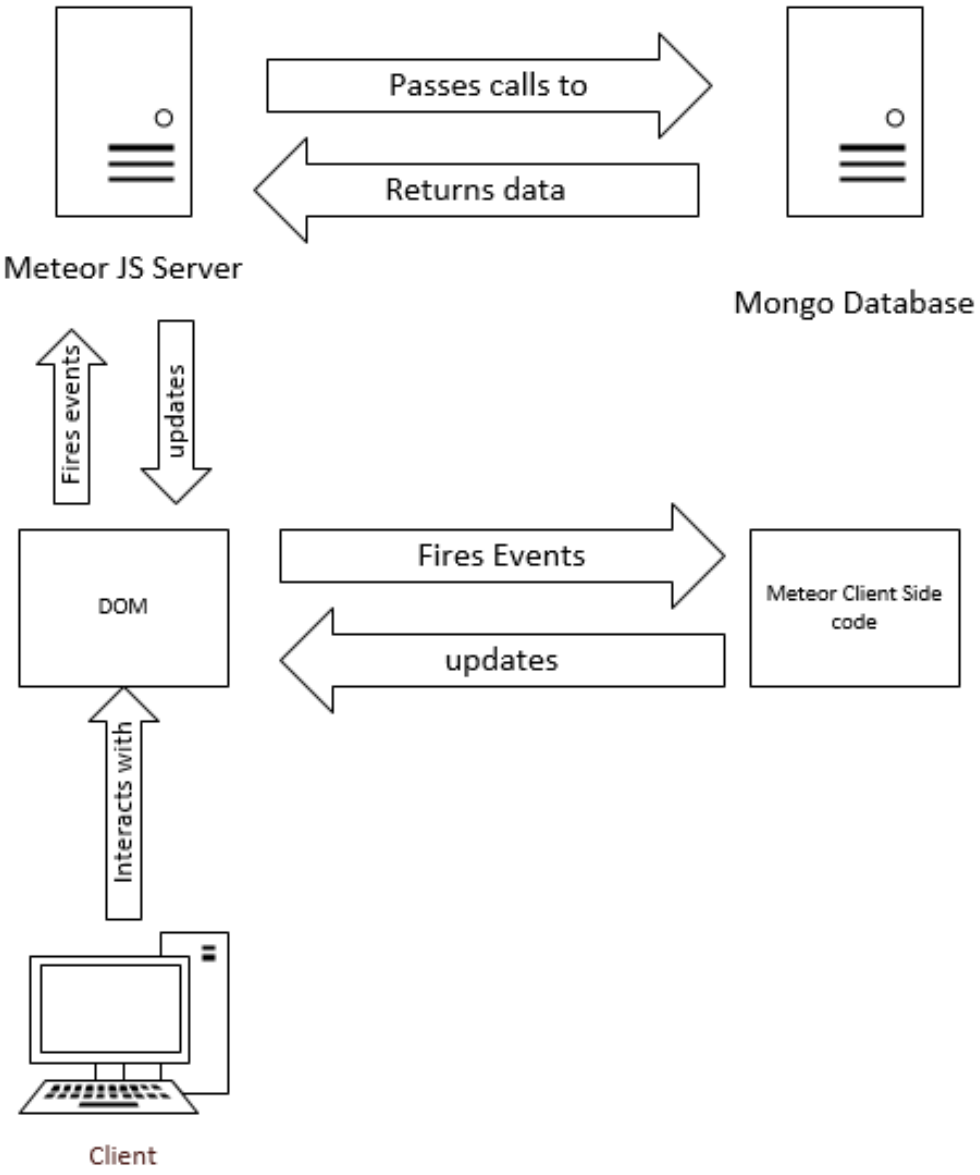


Figure 4: Technical Solution Diagram