# Hometown Heroes

CS 361 - 400, Group 6, Homework 6

Aug 8, 2017

Jon Austin
developer
austinjo@oregonstate.edu

Charlotte Murphy
developer
murpchar@oregonstate.edu

Valerie Chapple
developer
chapplev@oregonstate.edu

Gregory Niebanck
developer
niebancg@oregonstate.edu

Kenny Lew
developer
lewke@oregonstate.edu

Benjamin Rodarte
client
rodartec@oregonstate.edu

## ABSTRACT

In this paper, we describe the process of completing our assigned user story for this week, our final status, and our schedule for next week.

## INTRODUCTION

Hometown Heroes is a web-based app designed to increase volunteerism. Users can publish service opportunities, search for events, register for events, and participate in service events. A Hometown Hero is a user who volunteers his/her time supporting a Hometown Heroes service event.

The purpose of this document is to describe the first implementation of our system. This includes a link to our software and a description of how to install and use it. We will also describe the work we did on the user story we assigned ourselves this week, our interaction with spikes and UML diagrams, refactoring, integration testing, our schedule for next week and our interaction with our customer. We will also provide a burndown diagram of tasks completed.

## SOFTWARE REPO AND INSTALLATION

Our team used a GitHub repo for collaboration. It can be found here: https://github.com/vchapple17/hometown-heroes.git. To run the software, users must have Node and mySQL installed on the server. After the repo is cloned to the appropriate machine, users are to navigate via command line to the folder called `server`. In the `server` folder, the user must run `npm install` to download dependency packages. After dependencies are installed, the user starts the server by typing `node ServerCode.js`. To view the main page, open a web browser and go to http://localhost:16661/.

## USER STORIES DISCUSSION

Our work in HW5 showed that the first user story we should tackle would be #1, account creation via email address and two social media systems. It seemed logical that the first action the user must take to join Hometown Heroes would be the first to be implemented. Once accounts can be created, users can explore the other features of Hometown Heroes, and additional features can be added incrementally without disrupting regular user activities.

### Pair Programming

Programming pairs were divided by the following tasks:

- Valerie and Greg worked on full stack implementations of Social Media logins and account creation through Facebook/Twitter API
- Jon and Kenny worked on full stack implementation of manual account creation using an email address
- Charlotte worked on database implementation

## Unit Tests

*Frontend account creation* - Verified that each page could be accessed by a browser

- Link on homepage to sign-up page navigates correctly (status 200)
- Link on sign-up page to email account creation page navigates correctly (status 200)
- Link on sign-up page to Twitter account authorization page navigates correctly (status 200)
- Link on sign-up page to Facebook account authorization page navigates correctly (status 200)
- Twitter responds to Hometown Hero callback URL with Twitter ID saved in express-session
- Facebook responds to Hometown Hero callback URL with Facebook ID saved in express-session
- Homepage shows user specific information when user is logged in with express-session.

*Backend account creation with email address* - Verified that account information was inserted into the database

- Server accepts POST request from client account creation page containing new user account information.
- Server inserts new account information data into the database.
- Server sends data from POST request back to client.
- Client redirected to successful account creation page.

## Problems

During the past week, many of our tasks went relatively smoothly. However, there were some notable instances where implementation took longer than expected. First of all, most of the team members have never worked on GitHub in a group setting. While the team was able to push and pull from GitHub without issues, there was time required to research and understand the best way to utilize branches and merging. This was time not factored into the time estimates.

One unforeseen issue with User Story #1 was the extra requirement of utilizing Express-Session and Cookie-Parser for the Passport Twitter strategy due to its reliance on OAuth (not OAuth 2). Incorporating these additional modules was not difficult; however, the order of configuring the session with express, as well as with passport proved to be quite the puzzle. However, future implementations of the Twitter will be easier due to high reusability of code.

## Time Required

Below are the tasks set for User Story #1, as noted in an earlier homework. The times listed are the actual time required to implement. Note: 1 Unit of Time = 1 Programmer Hour

- iOS spike 5 units for research, 58-60 units for implementation (not implemented)
- Facebook spike
- Twitter spike - 3 units for spike, 5 units for integration and debugging
- Account Signup Page UI and routes - 2 units
- Frontend Account Creation with Email - 5 units
- Backend Account Creation with Email Address - 5 units

## Current Status

All three spikes for User Story #1 have been completed and evaluated for Hometown Heroes. Two of the spikes (Facebook and Twitter) easily were incorporated into the working app. The rest of the user story is mostly implemented and tested. That is, a user can navigate to a sign-up page, select a sign-up option (email, Facebook, or Twitter), and begin the account creation process.

## To-Do

Items not implemented currently include building the application programming interface between the server and the database. For example, a user database stub was created to implement the Twitter login with express-session, but no persistent storage was utilized. For full implementation of both Twitter and Facebook, an actual interface needs to be created to connect the server to the database so that a user can be saved based on Facebook ID or Twitter ID. Additionally, login functionality via email is not yet implemented. This is a prerequisite to next week's tasks for those users who use the app with email.

## SPIKE AND UML DIAGRAM DISCUSSION

*iOS Spike* **-** This spike was helpful because it allowed us to consider how much time we would need to spend in order to develop this as an iOS app as originally intended, given that most of us have no iOS development experience. After performing the spike, it became apparent that it would take far too much time to produce anything workable to present to our customer in the time frame given.

*Facebook Spike* - This spike was helpful for two main reasons. First, it allowed us to determine how much time was required for implementing social media as a way to log in and sign up, which is a functional requirement for the project. It also led to the discovery of some versatile middle-ware called PassportJS. PassportJS is middleware that works with sessions and NodeJS which provides authentication using Facebook and other social media integration as well. Discovering this middleware allowed for completion of additional user stories.

*Twitter Spike* - The Twitter spike was extremely helpful for implementing Twitter login. Once we determined that PassportJS was an appropriate tool to utilize in Hometown Hero, it was extremely easy to get the user to be redirected to Twitter, sign in, and be directed to the Hometown Hero callback page. However, the Twitter spike did not distinguish the difference between the Twitter implementation vs. the Facebook implementation. That is, the *passport-twitter* module required the use of sessions. A session manager was required for Twitter because the specific *passport-twitter* module operated on OAuth as opposed to OAuth2.

*Account Creation with Social Media Account* - The UML diagram for account creation through a social media platform was not very helpful because both Twitter and Facebook provide comprehensive APIs to use these features. Process-wise, it was fairly straightforward to implement.

*Account Creation with Email Address* - For the frontend of the email account creation task, the UML sequence diagram was not as helpful as the table of routes that were submitted in the last homework. This table allowed the team to ensure that the folders were correct for all team members and that the criteria for each page were met. The diagram was moderately more useful for the backend of the email account creation task. However, the task dataflow followed a basic sequence of steps of client to server to database, which is not complicated enough to need a diagram.

## BURNDOWN DIAGRAM

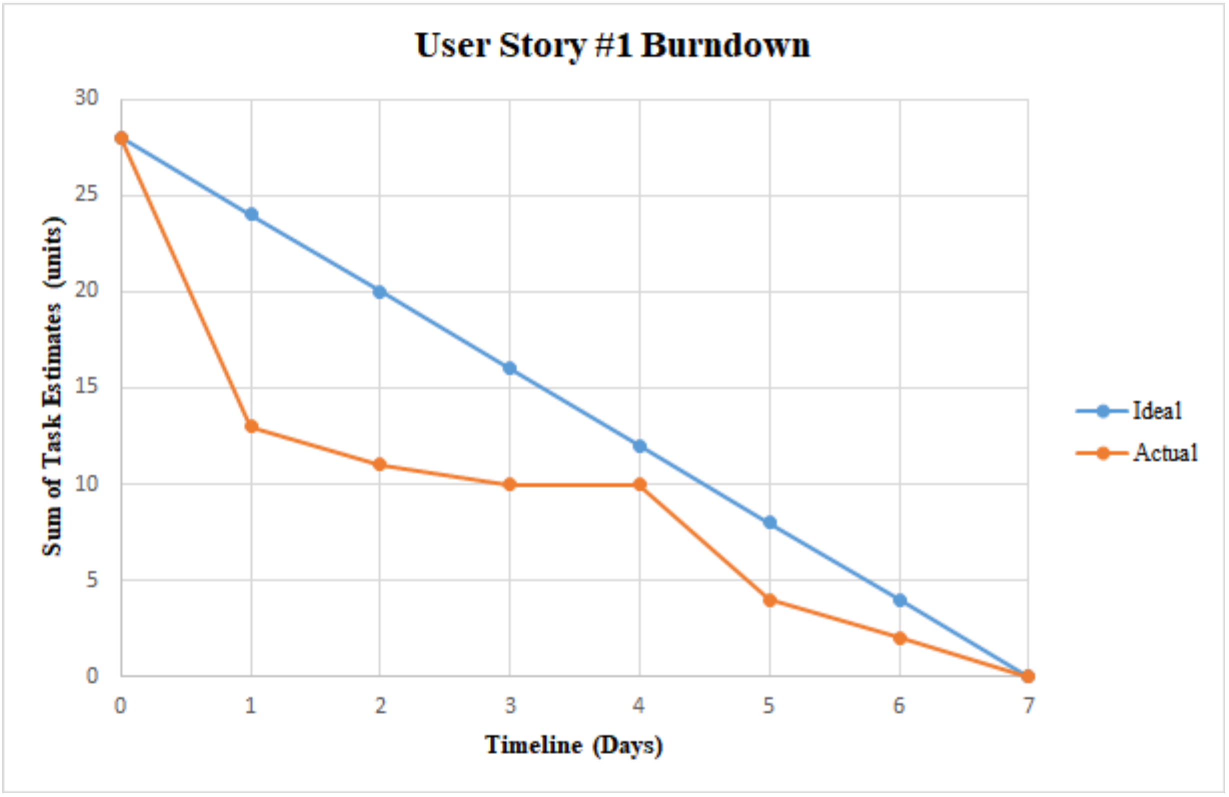

**Figure 1. Burndown Diagram**

## REFACTORING

The server code requires a great deal of modularization due to the numerous tasks that the server code will be implementing in just this week (social media authorization, account creation, database API, routes, views, and styling). However, when first beginning our product, we did not necessarily know how all of these items would relate to each other. Therefore, there were a great deal of opportunities for refactoring.

The first version of Hometown Heroes had a Facebook login setup with basic views and server configurations. However, all of the code for express and its dependencies was implemented in one file. The first refactoring was to modularize the routes into a separate folder that easily allows routes and sub-routes to be added, such as authorization and sign-up routes. Future routes may consist of routes to configure a user's profile and event management.

Another refactoring occurred after the social media login was integrated into Hometown Heroes. While two social media logins was not an extreme amount of code, it was easy to modularize these two setups into a passport folder. Also in this passport folder was a file to configure passport to utilize the appropriate strategies. Besides decluttering the main server file, this refactoring allows for other programmers to easily add other passport strategies - such as Google, LinkedIn or Instagram - into Hometown Heroes. This refactoring, along with the modularization of the routes, allows an easy addition of passport routes, views, and configurations.

## CUSTOMER QUESTIONS/REQUIREMENT CHANGES

We did not have to contact the customer this week. Our plan from last week was pretty solid. It allowed us to divide the work in a logical and effective manner, and we had clear and obvious tasks to complete for the week.

## INTEGRATION TESTS

The user stories chosen to completion in the first week involved the development of a web-based application that will emulate the appearance of a mobile device to handle account registration.  Hometown Heroes is required to allow access by creating an account through an email address, a  pre-existing Twitter and Facebook accounts. These units needed to be integrated into the server using the same routing index and middleware implementation. Individual units were tested with code adapted from our spikes, but because passportJS requires integration with express-sessions, much of that overhead code was duplicated between the individual units. For Facebook and Twitter, authentication access required registering domains with those services using their respective app development interface. This was best achieved by running our server application on the localhost, as opposed to creating a list of acceptable domains for which to register with our social media applications. Finally, all of these units represent the different methods for logging into our application, and thus will be accessible from the same page.

Once the routes and front-end code were integrated, navigation testing was done manually. Since we had done unit testing previously, we had to ensure that the functionality persisted after integration. The 'sign in' route renders a page that has buttons for all the different registration methods. In our case, navigating through the site, testing each of the buttons for desired functionality and expected outcome, as a user would have to do, was a sufficient integration test.

## NEXT WEEK

Next week we will implement user stories #2 and 4. The tasks required are shown below:

### *User Story #2*

As a first-time user, I can fill out a profile which includes the types of volunteering events I am interested in and the types of skills I possess so that the most relevant opportunities can be suggested to me.

1. Backend - API to read Interest Tags and Skill Tags (Time: 5 units)
2. Frontend - Form to collect user profile information, including Interest Tags and Skill Tags (Time: 6 units)
3. Backend - API to create new or update user profile (Time: 4 units)

### *User Story #4*

As a user, I can edit my profile and update my preferences/skills so they remain relevant.

1. Frontend - Prepopulate profile creation form, including Interest and Skill Tags (Time: 6 units / 3 units if reusing code)
2. Backend - API to create new or update user profile (Time: 4 units)

### *Jon Austin*

1. User Story #2, Frontend - Form to collect user profile information, including Interest Tags and Skill Tags

### *Valerie Chapple*

1. User Story #2, Backend - API to read Interest Tags and Skill Tags
2. User Story #1, Backend - API to create and retrieve social media user accounts

### *Kenny Lew*

1. User Story #4, Frontend - Prepopulate profile creation form, including Interest and Skill Tags
2. User login via Email

### *Charlotte Murphy*

1. User Story #4, Backend - API to update user profile

### *Gregory Niebanck*

1. User Story #1, Backend - API to create and retrieve social media user accounts
2. User Story #2 Backend - API to read Interest Tags and Skill Tags

## CONCLUSION

Our team completed our assigned tasks for the week. With good planning and constant interaction with each other as programmers, our programming tasks went fairly smoothly. We were also fairly close to our estimates. Considering these outcomes, we are fairly confident that we can make a plan and execute it with accuracy.

## CUSTOMER AVAILABILITY

We did not contact the customer this week.

## TEAM CONTRIBUTIONS

### Jon Austin

1. Front end email address account creation
2. Report Introduction
3. Collaborated on Installation Instructions, User Stories Discussion, Spike and UML Diagram Discussion

### Valerie Chapple

1. Passport-twitter implementation with express-sessions
2. Twitter spike discussion and Server skeleton
3. Collaborated on Refactoring and User Story discussion

### Kenny Lew

1. Back end email address account creation
2. Collaborated on User Stories Discussion - Unit Testing

### Gregory Niebanck

1. passport-facebook implementation
2. Facebook spike discussion
3. Main layout HTML / CSS to emulate mobile device screen
4. Integration testing

### Charlotte Murphy

1. Database implementation
2. Formatted document