January 2016

Admin Portal

Penelope Sarah Over
Worcester Polytechnic Institute

Ruofan Ding
Worcester Polytechnic Institute

Thomas W. Grimshaw
Worcester Polytechnic Institute

HackBox Admin Portal

Ruofan Ding, Thomas Grimshaw, Penelope Over

December 18, 2015

A Major Qualifying Project Report:

Submitted to the faculty of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

degree of Bachelor of Science

Sponsor:
Microsoft Corporation

Liaison:
Ben Fersenheim

Advisor:

David Finkel

ABSTRACT

This project was created to design and implement an Administrator Portal for Microsoft's new hackathon management tool. The Admin Portal provides administrators the ability to view, create, modify and delete hackathon related data. This data includes the basic information about a hackathon, the hackathon's participants, and the hackathon's projects. The Admin Portal allows administrators to easily perform these functions through a data centric user interface. We created the Admin Portal using the latest web development technology and with design inspiration from other major Microsoft products. We developed this tool in Microsoft's Cambridge office in collaboration with the Microsoft Garage team in Redmond.

ACKNOWLEDGEMENTS

We would like to thank Microsoft for sponsoring this project and providing us with the resources to complete it. We appreciate the challenging but meaningful project they proposed, and that it gave us an opportunity to both apply what we have learned at school in a software industry setting, and to expand our knowledge beyond our classroom learning. We would like to thank the Microsoft Garage team, especially Steve Scallen, Mahendra Reddy, Andrew Komar and Mike Pell, with whom we worked closely. We would also like to thank Ben Fersenheim and George Matthews, who helped and supported us over the course of the project. In addition, we would like to thank Henrik Joreteg and Jason Wilson from Formidable Labs for building the backend server our product connects to. Last but not least, we would like to thank Professor Finkel for his guidance throughout the whole project.

Contents

Chapter 1. INTRODUCTION

Hackathons are contests to design, create, and share computer programs in a specified time frame. Hackathons can last anywhere from a few hours to a week, with dozens to thousands of participants working on projects. Hackathons are designed to be places to share ideas, network, and learn new technologies. When a hackathon is organized, it tends to have a specific focus, whether that's centered on a specific language, platform, or cause, however not all hackathons have such a theme. In more recent years, hackathons have begun to include non-software related components such as hardware, civic engagement, and education. Many hackathons are sponsored by companies such as Microsoft. Microsoft has run many hackathons including the Oneweek hackathon which involves over 12,000 employees of Microsoft and generates software like TalkEasy, which helps those with hearing impairment[i], and Neuroversity which helps kids with autism[ii].

1.1 Background

While hackathons have been growing in popularity over the past decade, there are still some issues that participants face when attending them. Before the hackathon even starts there is the issue of coming up with a project to work on. Very few hackathons provide a method for participants to collaborate and discuss project ideas before the event starts. Having such a collaboration tool allows the hackers to spend more time at the hackathon working on the project rather than coming up with an idea. Additional communication prior to the hackathon would also help to form teams, allowing those with ideas to link up with those with needed other attendees before the event even begins[iii].

After arriving at the hackathon, one of the first problems participants may face is the amount of time it takes to setup their development environment. Themed hackathons may be working on a specific piece of technology which requires certain languages, IDEs, or SDKs that attendees may not have installed. Getting all of these setup can take up to a day and detracts from the productivity of the hackathon's participants. Having an automated method of providing development environments that are already setup helps to improve the hackathon for the participants, by allowing them to get going on their projects faster.

In order to help solve some of these problems, Microsoft has been developing their own hackathon tool to assist in the organization and execution of hackathons. The hackathon tool was originally created to support Microsoft's Oneweek hackathon. The tool is designed to handle participant registration, as well as provide teammate and project search. In HackBox, when participants create a project there is an option to automatically provision a Visual Studio Online repository so the team can hit the ground running and not have to deal with setting up their environments[iv].

The previous version of Microsoft's hackathon tool, called HackBox at the moment, was created by modifying a third party conference scheduling tool. As Microsoft is looking to further develop HackBox into a service Microsoft can offer to other organizations, they have to redo much of the backend of the current tool, which is owned by another company. The three main components that must be redone are the database, front-end website, and the administrator's portal[v].

1.2 Project Goal

The goal of this project was to develop a new administrator's portal for Microsoft to use in their new version of HackBox. The Admin Portal has the ability to manage the various hackathons, projects, participants, and logs that are part of the HackBox system. The design for the Admin Portal has been updated from the previous version to reflect a more modern layout that offers enhanced functionality. The design of the Admin Portal is based on the new Azure Portal and various other Microsoft tools. The new Admin Portal we built uses the idea of blades from the Azure Portal, where a blade is a panel of more detailed information sliding out to the right, when something is selected.

Chapter 2: LITERATURE REVIEW

2.1 Existing Hackathon Platforms

There are some other hackathon tools that already exist, such as Hacker League and Hack Dash. Hacker League provides hackathon organizers with online infrastructure and a comprehensive suite of tools to organize a hackathon event[vi]. The platform is used for registration of participants, and is more convenient for organizers to get participants' information and manage hackathons. It can also be used as a place to post announcements for participants and journalists so that they will get the latest updates about the hackathon events. It is not only for hackathon organizers but also for hackathon participants. Hacker League provides participants a place to share their thoughts and collaborate on their work. It allows users to comment on submitted hackathon projects, and give suggestions for future improvements. By looking at previous hackathon projects, participants can be inspired to create their own projects. Also, participants can upload their hackathon projects for others to view, which makes it easy for them to collaborate.

By contrast, the Hack Dash platform is solely for hackers to share ideas for a hackathon[vii]. It is a community where users can upload their projects and views others' projects. The platform also allows users to follow, join and comment on each other's project.

Wolfram, also, has its own development platform, called Wolfram Development Platform, which is very different from other hackathon platforms[viii]. It is a cloud based platform, and has many functionalities that are useful for hackathons, including visualization, image processing, instant API creation, and automatic Java & .NET Connectivity. Since Wolfram's platform has already generalized various technologies into its own Wolfram Language and is completely cloud based, it makes environment provisioning simple for hackathon participant.

2.2 HackBox

The HackBox system was originally designed by Microsoft's Garage[ix] team to support Microsoft's Oneweek hackathon. Before the event starts, HackBox handles the registration of thousands of participants and hundreds of projects. With the previous version of HackBox, participants could specify skills that they have and search for teammates and projects based on

skills, location, and other search parameters. HackBox is also able to automatically provision a Visual Studio Online repository for projects and record the winners of the hackathon.

While HackBox already has many similar features to its competition, HackBox is currently only used internally to Microsoft. In order to facilitate the use of this product outside of the company Microsoft must remake much of the codebase as the current HackBox is built on a platform owned by another company. As part of the recreation of HackBox, Microsoft must recreate the database and its API, and the Admin Portal.

The previous Admin Portal allowed the admins to instantly add participants and projects, as well as generate logs of everything that occurs on the HackBox system. The new Admin Portal has all of these features, as well as some new ones. First the new portal has the ability to manage multiple hackathons each with multiple projects. Another improvement is an updated user interface with inspiration based on the new Azure portal that has received praise for its design[x].

Chapter 3: METHODOLOGY

For this project, we worked with a team in Microsoft's Garage from Microsoft's Redmond Headquarters. As this team was located on the other side of the country from where we were working, most of our communications were over Skype and email. In addition to the Garage team, we worked with contractors who were working on creating a database server for storing the hackathon information and serving it up over an API.

Before starting the project in October we met with the Garage team and discussed what the project would entail. The goal of the project was to develop a single-page web application that had all of the functionality of the old Admin Portal, along with the ability to handle multiple hackathons. The old functionality involved being able to add, delete, and edit projects and users.

3.1 Development Environment:

The development of the Admin Portal was done using several tools. These tools were selected based on their appropriateness given our project's guidelines and requirements.

3.1.1 Visual Studio

Visual Studio is an IDE developed by Microsoft to allow developers to write many different types of programs. We chose it specifically for its integration with Node.js[xi] which we used as the framework for our project. We used Visual Studio to write all of our code and tests, from HTML, to Sass and JavaScript. We also used Visual Studio to run a local Node.js server for testing. Using Visual Studio to run this local server allowed us to debug our code when problems occurred. On top of the basic editing and debugging functionality, Visual Studio was also chosen for its impressive number of extensions including some recommended to us by the Garage team. One of the extensions recommended to us was Chutzpah[xii] for running our JavaScript unit tests in Visual Studio. Along with basic testing, Chutzpah also allows us to perform code coverage analysis.

3.1.2 Git and GitHub

Git was our source control management (SCM) system of choice for a variety of reasons. First of all, Git is a modern SCM that allows for easy branching and merging of work[xiii]. Secondly it allowed us to use GitHub as our source code repository. GitHub is a website that allows team

members to host Git repositories[xiv]. The GitHub repository we used was part of the Microsoft group on GitHub. This allowed the repository we used to be private and only accessible to Microsoft Employees. Another reason we chose Git and GitHub was that the entire team had experience using them before.

3.1.3 Browser Developer Tools

Many modern browsers include tools, often called developer tools, for those developing websites. These tools include the ability to inspect html elements, step through JavaScript functions, and much more. We used these tools to debug our web app over four different browsers, Internet Explorer, Edge, Chrome, and Firefox. We tested across these browsers because the Admin Portal is being developed for Microsoft employees initially, so Edge is going to be one of the most popular browsers used, and the other browsers are all popular on Windows systems. The developer tools were especially useful for debugging our CSS code and determining cross browser compatibility on Internet Explorer 10. Our primary development browser was Chrome because its developer tools had the most features, and the ability to be in window as well as in a separate window tool.

3.2 Technologies

3.2.1 AngularJS

AngularJS is a web application framework which provides client-side model-view-controller (MVC) and model-view-view model (MVVM) architectures[xv]. In order to be used by a web page, the AngularJS library needs to be included in the HTML page as a script, and will compile and render angular custom tag attributes into Document Object Models (DOMs).

The first reason we used AngularJS was that we were building a single-page web application. This web application needed to display data resources dynamically and be able to respond to user actions in a single page. The traditional single-page web application uses server-side model-view-controller architecture. For example, an AJAX request is sent to the server to ask for some new content; the controller in the back-end manipulates the model based on the request, and updates the view (front-end HTML) by sending newly rendered HTML to the client. In this process, because the controller is at the server, most of the data processing and all HTML renderings happen in the back-end. In contrast, AngularJS provides a

client-side MVVM mechanism, allowing for two-way data binding. AngularJS' two-way data binding binds DOM elements (view) and the actual data (view-model) together. Updating either of the two will update the other as well. Therefore, all of the HTML rendering is on the front-end. Indeed, AJAX calls are still needed to retrieve data from the server or update data in the database, but the workload on the server is significantly reduced and the web app responds to changes faster by using AngularJS, which is due to the reason that view-model can do its logic operations and change view all on the client side.

The second reason we chose AngularJS is that it provides testability. Unlike JQuery[xvi], Angular decouples DOM manipulation from app logic. AngularJS's directives and controllers are separated from DOM, and scope, an object that reference to the application model[xvii], is used as the glue between the application controllers and the view. Because of this separation between view and application logic, all controllers and directives are testable individually. Unit-testing and mocking can be easily applied to our tests because of this separation.

Finally, AngularJS is maintained by a huge developer community. There are many open source libraries and resources designed to be used with AngularJS that we were able to use for this project. For instance, we used Angular Material for some of the Admin Portal's UI components.

3.2.2 Azure

Azure is Microsoft's cloud computing platform designed to host multiple different types of projects, languages, and operating systems[xviii]. We chose Azure as the host platform for our app not only because it is a Microsoft product, but also because it makes it easy, and quick to deploy and test our web app. Azure has the ability to tie into source control management platforms like GitHub and pull and deploy the master branch automatically when there are changes. This makes deploying to Azure as simple as pushing the master branch of our project. We also chose Azure because it has easy integration with Active Directory, an easy to use authentication method. On top of using Azure to host our web app, we also used Azure as the inspiration for the blade design of our app.

3.2.3 JavaScript

JavaScript is a high level language that can be used in many different ways, the most common of which is for the development of websites[xix]. One reason it is good for web development is that it is interpreted instead of compiled, allowing it to be changed quickly and be easily redeployed. Since we wrote a web app, JavaScript was the natural choice to do more advanced things such as sending http requests. One of the nice things about JavaScript is that it is so popular that developers have written lots of advanced libraries that others can use. We used a few different JavaScript libraries in this project, from the server running in Node.js to populating the data from the database into the HTML via AngularJS. JavaScript is supported on most browsers and most of the popular libraries like AngularJS are written to reduce compatibility issues. There were a few issues with backwards compatibility since several of the libraries we used depend upon features that did not exist in earlier versions. We ended up setting a minimum supported version of each major browser, and assumed that the other versions were too out-of-date to be popularly used.

3.2.4 Node.js

Node.js is an open source runtime environment for developing server-side application in JavaScript. It provides an event-driven architecture, and asynchronous callbacks are largely used in Node.js applications. The event loop of Node.js is that Node.js listens for events, and then triggers a callback function as an event happens. One of the most significant characteristics of this event-driven architecture is that a Node.js program runs in a single thread. Because there is only one thread handling all events, the program won't have any overhead of concurrency issues. This also means that no locking is needed, which is a time-consuming but indispensable component in common multithreaded architectures. A single-thread program also provides better scalability compared to a traditional Apache server[xx]. An Apache server will take up much more system resources as the number of requests increases, because each request needs to be handle by its own thread. Therefore, the event-driven architecture can boost the performance and scalability of a server application.

Another advantage of using Node.js is that it shortens the development cycle. Node.js provides most of required APIs for server development, including I/O operations and more. In

addition, Node Package Manager (npm) gives us access to various useful open source libraries. For instance, Express is used as our web application framework. The express framework allows us to configure the routes and add route handlers easily. By using express, we were able to set up our server and have it ready to go in only a few days.

3.2.5 Sass

Syntactically Awesome Stylesheets (Sass) "is an extension of CSS that adds power and elegance to the basic language"[xxi]. We chose to use Sass over basic CSS because Sass has variables, imports, inheritance, and allows nested rules. Sass also has the advantage that CSS code is valid SCSS code, which is not true for style languages like Less[xxii]. Using Sass also allowed us to keep our code clean and readable by nesting class definitions and using variables for things like colors and fonts. This also made the code easier to maintain as changing a font or a color only requires changing a variable rather than hunting down all of the instances of a color and changing each one individually.

3.3 Development Process

3.3.1 GUI Development

The GUI development for this project was done primarily based on mockups given to us by one of the designers on the Garage team. After receiving these mockups, we worked to implement the style and functionality displayed in the drawings. While the general theme, color palette, and broad functionality were determined by the team in Redmond, we implemented much of the functionality based on our own interpretation of the mockups. Once we finished implementing a set of features we would demo the results to the Garage team and they would suggest changes and additions for us to make.

There were a few design principles that we took into consideration when making our user interface. One of the big design steps we took was to introduce plenty of whitespace between the elements in an individual blade. This helped to reduce the feeling of clutter and to give the UI a cleaner look. In addition to adding white space we made sure that controls for each blade were clearly labeled, and used symbols that are standard for their associated actions. Another big UI theme was keeping the interface consistent across multiple different contexts. The blade

designs for both participant and project have similar buttons and layouts so as to give the user a feeling on consistency and continuity when using the app.

3.3.2 Database Integration

The Admin Portal was designed to interface with a separate data source for information such as users, projects, hackathons, etc. At the beginning of the project this data source was a csv file of the previous summer's Oneweek hackathon. Using this data source required writing a csv parser and providing a RESTful API in front of the parser so that the web app could access the data via an AJAX call. This temporary solution allowed us to develop the Admin Portal while the actual database and API were still in development.

Once the Redmond team had the database mostly set up they had us migrate the data from the csv to the new database via the API. To perform the migration, we wrote a Node.js program to translate the csv file to the proper JSON format. This migration discovered some issues with both the API and the database behind it. There were, with the first database, some issues with the number of connections between the API and the database causing the server to return 500 or 503 errors on the majority of requests. This caused our migration to progress slowly as the server could only handle 4 requests simultaneously. There were also problems with the size and configuration of the first underlying database. We communicated these issues to the Redmond team and worked with them to resolve the issues and get the data migrated to a new more robust database. After we finished migrating the data to the new database, we changed our AJAX calls in our web app to point at the database server's APIs rather than our server's APIs that served up data from the csv.

Chapter 4: RESULTS

4.1 Code

We delivered a single-page web application in AngularJS. The Angular web app allows hackathon administrators to view, add and modify hackathon related data. It uses ajax calls to connect with HackBox's RESTful server for retrieving and updating data. The web app uses Active Directory (AD) to authenticate the users before they can use the web app. It also uses Azure's Graph API to retrieve user information such as profile pictures.[xxiii] The web app supports most popular browsers, including Internet Explorer 10+, Edge, Chrome, and Firefox.

The source files of our web application are the main delivery of our project. The JavaScript of our Angular app is split into different modules. The three components of these modules are controllers, directives, and services. First, the controller modules have all our Angular controllers that contain business logic and are attached to the DOM of our HTML content. There is the *bladeController module, hackathonController module, mainViewController module, userControllerModule, and projectControllerModule*. Second, the *service module* provides services commonly used across the app, such as a RESTService to communicate with the database, and GraphAPIService. The *service module* organizes common services and improves the reusability of our code. Third, the directives module contains customized HTML tags with desired functionalities. For example, the lazyloadingTable is a customized HTML tag that is used to display the list of projects or users. The modularity makes code more readable and extensible.

We also delivered the HTML and Sass files that make up the UI part of our project. These files have been broken up by section, and are designed to be easily extensible for future developers of the project. One of the other ways we have made our project more extensible for future developers is by having test files for the JavaScript in our app. The test files contain almost 80 test cases. At completion of this project all of our test cases are passing, allowing these test cases to be used for regression testing in the future development.

This single-page web app is running on a web server in Node.JS that can deliver the HTML pages, style sheets, and scripts of our web app to clients. The Node.JS server uses the Express

framework. It is currently deployed on Azure, and can easily be deployed on any other web hosting server.

4.2 Features

The HackBox Admin Portal delivers the ability to manage each of the major pieces of data that a hackathon contains, and does so through an interface that is clean and easy to use. Each of the features we deployed to the web app has an accompanying Graphical User Interface (GUI). The GUI is the way the user interacts with our web app and as such is very important. The HackBox Admin Portal GUI contains several pages or blades that make available features for the start page, the Hackathon information, the Project information, and the User information. The blades are designed after the Azure portal with blades opening to the right and scrolling left to right instead of up and down. Each blade contains more refined information than the blade to its left.

4.2.1 Start Page

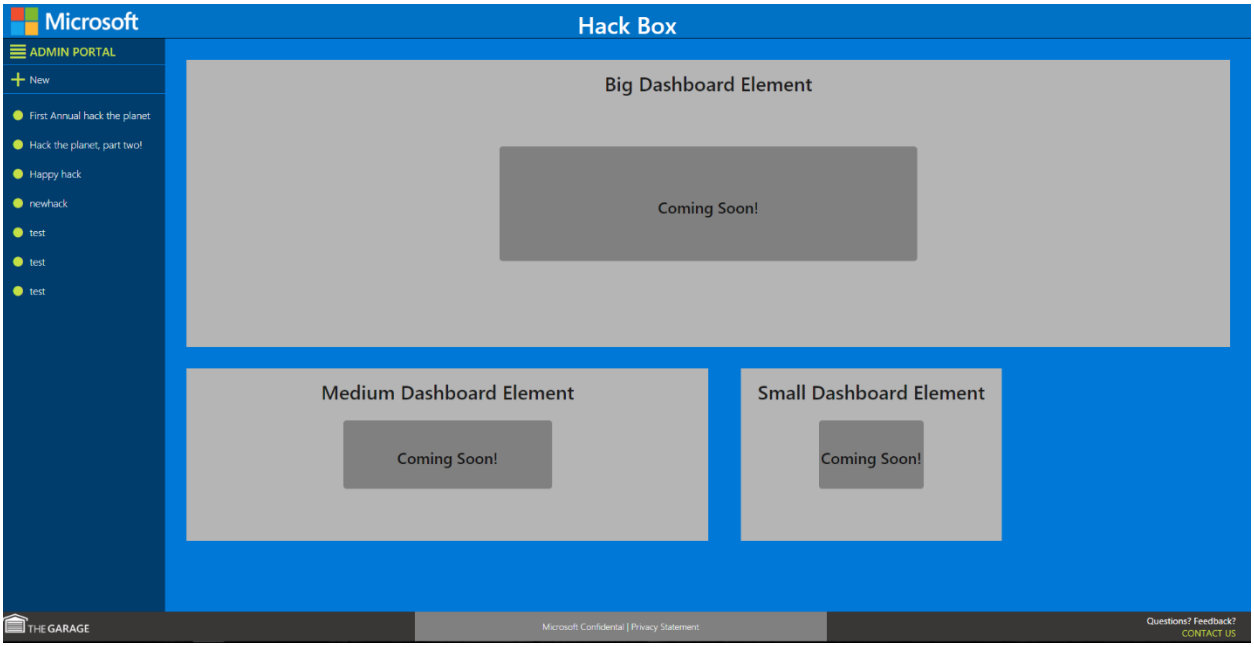

Figure 1: Start Page

The Admin Portal web app has three permanent elements as can be seen in 'Figure 1: Start Page': the header, footer, and sidebar. The header simply contains the Microsoft logo, the HackBox title and space for a logo when one is created, and room on the right for information about the logged in user. The footer is similar but has a few more elements. It is split into three

sections, one each for the Garage logo, the confidentiality information and a contact us link. The outermost thirds are dark gray and the innermost one light gray. The footer is the only piece which is the same on the front end web page and the Admin Portal. The sidebar contains business logic, displaying links to each Hackathon and a button to create a new one. When a user clicks on a hackathon and opens new blades to the right, the hackathon will stay highlighted as a reminder to the user which hackathon they are editing. At the top of the sidebar is the title Admin Portal in contrasting colors to the rest of the sidebar.

The start view also has a dashboard in the middle. The dashboard is the default view of our web app, currently displaying several blank boxes shown on a blue background. This space can be used for statistics, shortcuts, and other quick access information in the future.

4.2.2 Hackathons

Through the Admin Portal, admins are able to edit the basic information regarding each hackathon. This information includes items such as the title of the hackathon, the location, the time of the event, and the administrators of the hackathon.

Along with editing the basic information of the hackathon, the Admin Portal we created also allows admins to edit four of the six sub-sections of a hackathon: About, Projects, Hackers, and Custom Questions. Under the About page, admins are able to edit the description, schedule, rules, judges, and contacts of a hackathon. This section allows the admins to edit the data that appears on the main page of their hackathon, thus informing the hackers of the relevant information.

When a user clicks on a hackathon, the first blade that opens is the options blade, replacing the dashboard. The blade is the same width as the sidebar and contains links to the primary blades for that hackathon. The button for each blade has an icon and the name of the blade. The title of blade is the hackathon name, in white on a navy blue body.

The HackPage is the first button on the options blade. The Hack Page has the same header as the options blade including the save and publish buttons below the title, in lime green. The save button will be greyed out and disabled when there are no changes to be saved, or when there are invalid inputs in the required fields. The publish button can be clicked

whereas the opposite will be true when there are changes to be saved. The HackPage has several fields outlined in grey with red asterisk next to the required fields.

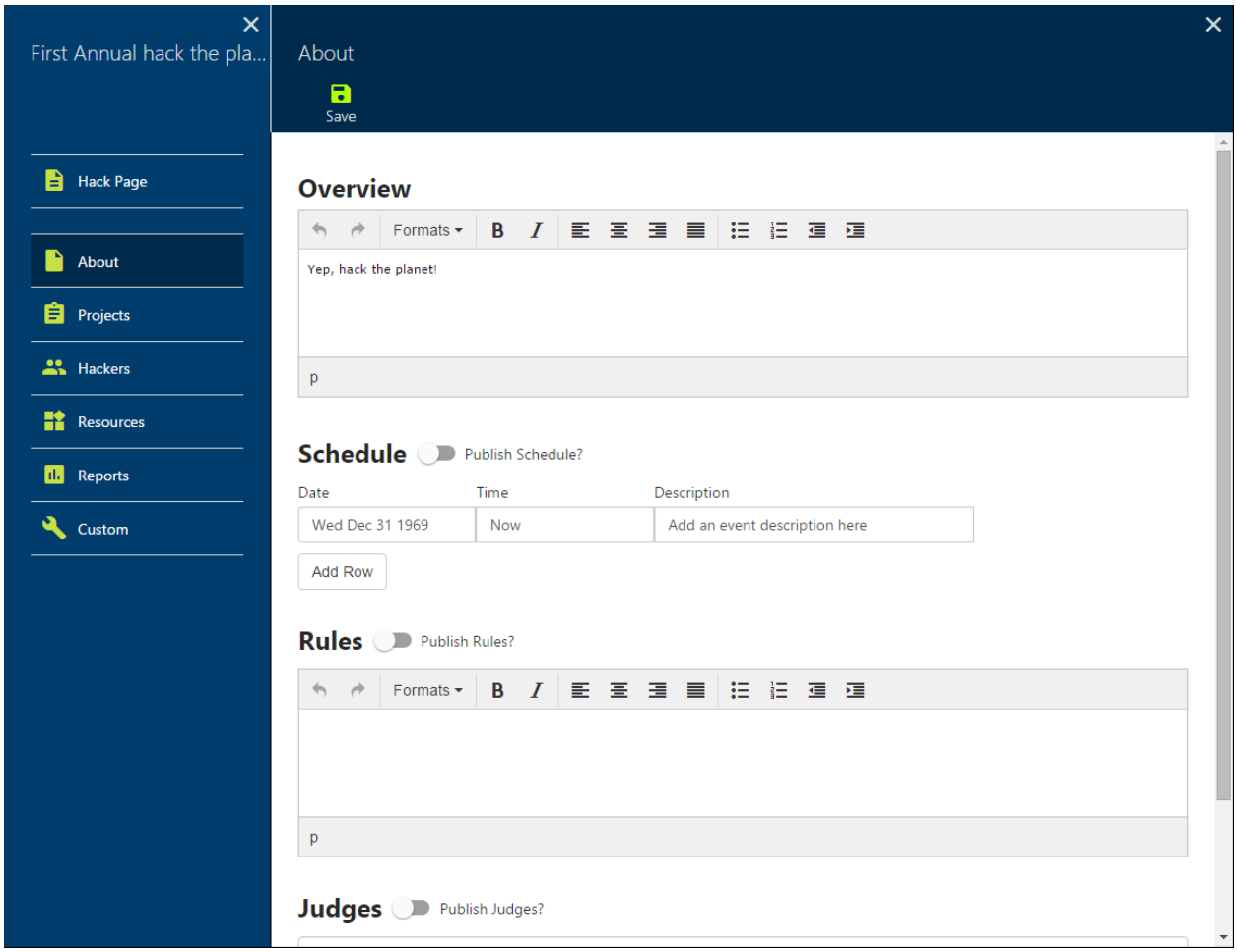

Figure 2: About Page

The About blade is very similar to the HackPage blade, with the rest of the general information about the hackathon, such as rules and judges (Figure 2: About Page). The about blade also contains the schedule, split into four columns, for date, time, day of the week, and description. If all of the existing rows are filled in, then the new event button will let a new row be added. A row can also be deleted if the event will no longer take place. The judges, rules and most other fields on the About Blade are not standard Bootstrap inputs, but an in-browser text editor called TinyMCE[xxiv]. TinyMCE gives the user more tools and space to format data to be displayed than an input would.

4.2.3 Projects

The Admin Portal also allows the admins to edit the projects that are a part of their hackathon. This includes the ability to add and remove projects, tag projects, and edit the projects information.

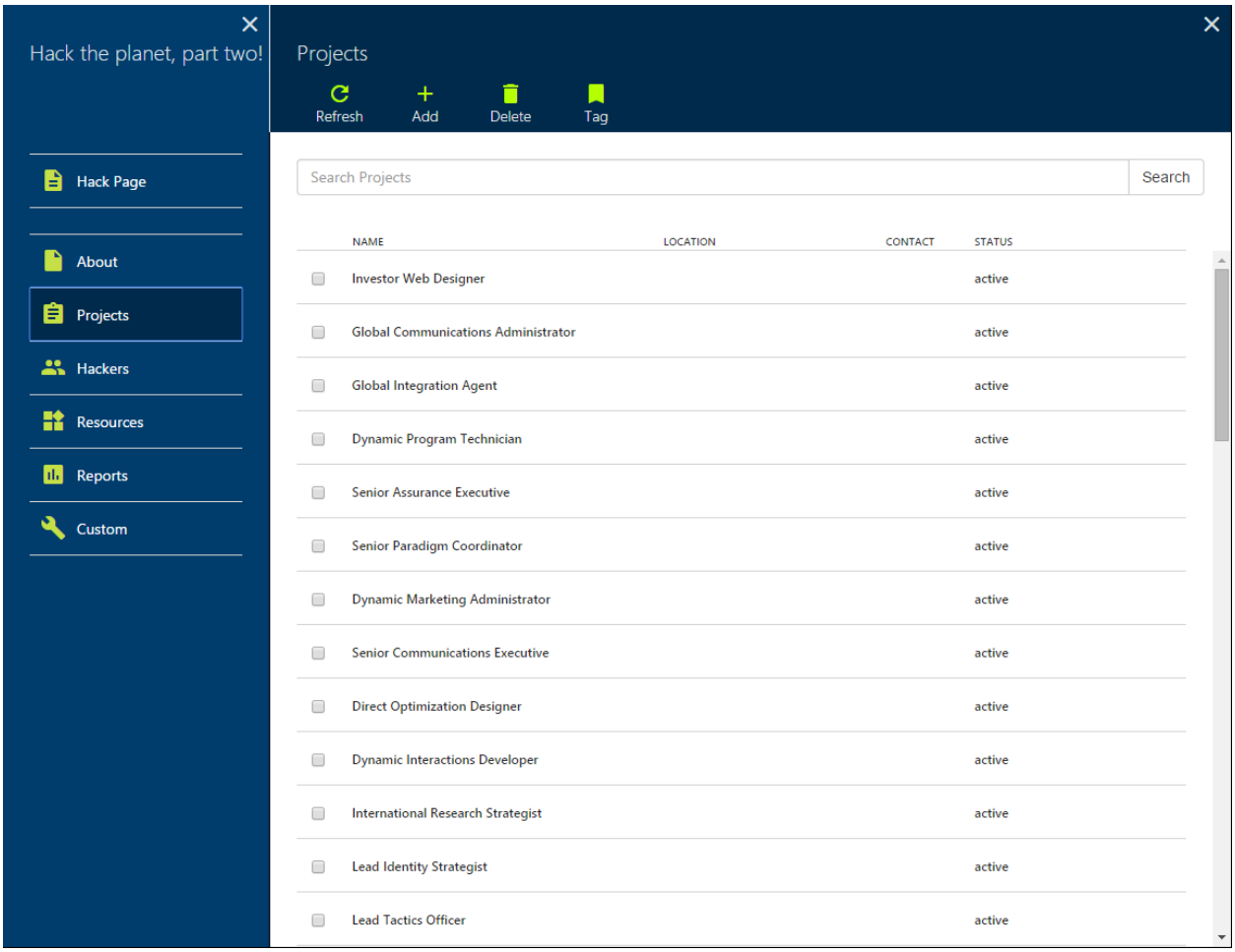

Figure 3: Projects

The Projects blade shows the list of projects, allowing admins to see what projects are a part of their hackathon as shown in 'Figure 3: Projects'. Above the list is a search bar with a button at the end to restrict contents of the list. In the toolbar there are buttons to create a project, delete a project, tag a project, and to refresh the list.

The individual Project blade has fields for the information relevant to the project. The title of the project blade is the name of the project, making it clear what you are editing. As there are so many fields for the project is has to have scrolling enabled so that all of the information

can be displayed in a clean manner. Only the blade in question will scroll not the whole page, allowing the users to have a clean and intuitive interaction with the Admin Portal.

4.2.4 Users

Along with projects, the admin can also manage hackers who are a part of the hackathon. The admins have access to change who of the registered users on HackBox are registered to their hackathon. This allows the admins to have control over the accessibility of their hackathon.

Much like the Projects blade (Figure 3: Projects) the Users blade provides the ability for admins to see all of the users in a hackathon. The admins can use the search bar, add participants to the hackathon, and tag users all from the list view. The individual Hacker view is similar to the individual project view as well, but with fields like name, skills, and location. The User blade contains many disabled fields, as the admins should only be able to edit the participant meta data related to the hacker partaking in their hackathon.

Chapter 5: FUTURE WORK

5.1 Admin Portal

The work done for this project was focused on creating the framework for the entire Admin Portal and providing full functionality for the majority of its content. While the features currently implemented make the Admin Portal a useful resource, there are three aspects that could be expanded upon. The first is the additional content that is needed in the Resources, Reports, and Dashboard sections of the portal. The second area of improvement is the design of some of the existing sections. And finally, while much of the Admin functionality has been implemented, the portal still needs the Super Admin functionality added.

5.1.1 Additional Content

One of the sections that the Admin Portal still needs is the Resources section. This section would be devoted to managing external resources for the hackathon. These resources would include things like code repositories, continuous integration, and any other resource the hackathon needs. Adding this content goes beyond the original implementation of the Admin Portal, so this requires more than just porting the old functionality. This section will be more than just listing information, as it will be used to actually manage these external resources. That means that each resource needs to be hooked up the Admin Portal by its API, allowing it to be managed from there. For example, if Visual Studio Online (VSO) was an external resource for a hackathon, then the Admin Portal would need to manage the subscription that the repositories would be provisioned under.

One of the sections that was in the original Admin Portal, but did not make it into our version was the Reports section. This section allows the admins to easily retrieve information on a given hackathon such as statistics about skills, average team size, average projects per participant, et cetera. This information gives the admins insight into their hackers and the projects. This section requires there to be methods to track number of views, number of hackers per project, et cetera, in order to easily summarize this information for the admins. This means that this section requires not only displaying existing information, but in order for it to provide maximum utility, also adding new data points to the dataset so that more information can be tracked.

Although the existing sections of the Admin Portal, and those discussed above, deliver the full functionality of the Admin Portal, having a view that summarizes select information in one place would be a big benefit to the portal. This information could go on the dashboard of the Admin Portal, allowing admins to pin select stats, lists, and projects to the start page of the portal, for easy viewing, and fast navigation. This customizable dashboard concept would be very similar to the Azure portal dashboard[xxv].

5.1.2 Design Finalization

Another area of future work would be for a designer to look over our portal and make tweaks where necessary. The existing portal had input from designers, but before HackBox is released to the public, more work should be done to clean up the user experience. Usability tests should be conducted to make sure that the design flow is logical. These tests would involve having people who have not used the Admin Portal before attempt to perform the actions of a hackathon admin, and record how easy it was for them to complete the task. This information would be gathered by observations and by asking the user what could be better after the fact.

5.1.3 Super Admin Features

The final area of future work in the Admin Portal itself would be the addition of super admin functionality. This functionality would include viewing all users in a single list, the ability to manage all hackathons, and the ability to edit a user's information. These abilities go above those of the standard admin and would be available to the team that manages HackBox, and would be used to help prevent abuse of the platform. The all users section would require the introduction of a layer above the list of hackathons as the list of all users does not belong under any one hackathon. This information could be placed on the dashboard, or under a menu somewhere.

5.2 HackBox Website

The HackBox front end website interacts directly with regular users for registration, hackathon sign-up, and project creation, while the HackBox Admin Portal serves as a backdoor for admins to check and modify hackathon related data. Both the website and the Admin Portal are connected to the same database, and most of data that the Admin Portal populates from is

inputted by users through the HackBox website. In order to create the full experience of HackBox and allow admins to be able to use Admin Portal to its full potential, the work on the front end website must be completed as well. In addition, the Admin Portal and the front end website need to stay consistent with each other. For instance, they should have exactly the same data validation for the same data schema. If one fails to filter out invalid data before saving it to the database, the other may fail to display the data correctly.

5.3 Database Work

5.3.1 Schema Changes

In the beginning of this project, we were mainly interacting with data from a csv file of 2015 Oneweek hackathon, and using the schemas from this csv file. However, as an API server and a brand new website were being built up from scratch, the old data schemas doesn't satisfy all the requirements anymore. The schemas of most datasets we were using, such as the project schema and the user schema, had been changing, so the HackBox Admin Portal kept changing in order to match the schemas. Since not all schemas have been finalized, and it is suspected there will be future changes to the schemas, our HackBox Admin Portal needs to be changed as well. Updating schemas will necessitate changes in both the existing HTML pages and JavaScript files, as fields names changes or fields are added or removed.

5.3.2 Data Migration

The data from 2015 Oneweek hackathon has not been migrated to the API server that both HackBox website and HackBox Admin Portal are connecting to. Hence, there is a need to migrate data from the csv file to the server eventually after all schemas have been finalized. We wrote scripts in Node JS for project data and user data migration, based on contemporary data schemas. These scripts needs to be modified to match the finalized schemas before being used for data migration. We also found an issue that there is more than one entry for the same user in the *MasterRegistrationReport.csv*, the list of users from the Oneweek 2015 hackathon, so duplicates elimination is necessary before data migration.

5.3.3 Database Stability

So far, the HackBox Admin Portal has occasionally had HTTP 500 Internal Server Error issues. This is due to the fact that the current data server we are using is not stable enough for

what we need it to do. In order to provide better user experience in the future, we should change the current data server to a more stable and scalable one.

5.4 Externalization Work

Before HackBox can be rolled out to the world there are several steps that need to be completed first including internal and external trials before it can finally be rolled out to the public.

5.4.1 Internal Trials

Once HackBox and its Admin Portal are complete, they will be released to Microsoft employees to use within their product groups or across the whole company like the Oneweek hackathon. Through these internal trials the Garage team will be able to find any problems that arise from deploying to a large number of people, and will be able to gather feedback from the users to make improvements or fix bugs.

5.4.2 Authentication Mechanisms

Currently the HackBox and HackBox Admin Portal use the Active Directory Service to authenticate users to the site. The Active Directory is a service that stores information about users or a large number of other things on a network.[xxvi] In this case, we are using Active Directory to make sure only users that are on the Microsoft Corporate Network can access HackBox and HackBox Admin Portal. While this method will work for organizations that use Active Directory, this will not work for organizations that use other authentication methods. This means that when this product gets rolled out to corporations other than Microsoft, other methods of authentication will need to be added.

5.4.3 Public HackBox

Once HackBox has been deployed to and used by external corporations, it is hoped that there will be a publicly available version of HackBox. This will be slightly different from the HackBox we have today. It will need to use external accounts like GitHub, Facebook, or LinkedIn to authenticate users to the main site and the Admin Portal. This also means that the public HackBox instance will need to be accessible from the internet rather than constrained to the corporate networks like it is now by Active Directory. This is important because we want the world, not just corporations, to use this tool. This can be achieved by the Garage team hosting

an instance of the tool that is publicly accessible, so smaller organizations and groups can use HackBox.

5.4.4 Product Name

A problem the Garage team has had throughout the project, is naming the tool we are working on. We are calling it HackBox until we find a name that we like and that passes the rigorous name check system by Legal and Corporate Affairs. We, our team in Cambridge and the Redmond team, have suggested several names, all of which were either already in existence or had a word or something similar. So, the tool's title, when it gets released to the world, will be different than it is now. We still don't know what it will be, just that it will not be called HackBox.

5.4.5 Globalization

Another issue we identified was that our tool does not support multiple languages. Even when this is released within Microsoft there will hundreds of people from other countries using it. Since there will be users who either don't know English or prefer another language, support for multiple languages will need to be added, which is a significant amount of work. This will require a changes to the Admin Portal, and likely the database and the front end as well. This is not a necessary part of the tool, but it will expand the user base and popularity of the tool significantly.

5.5 Generic Work

5.5.1 Personal Hackathons

There are still more features that the Garage team wants to add to the final project, the newest of which is to add personal scratch hackathons. Each user will have a personal scratch hackathon to jot down ideas or start working on a new project when they are not at that moment part of a hackathon. This will require changes in both the front end and the database. The front end changes would include the ability to see personal projects and add new ones. The database will need some changes so that the personal hackathons will not show up in the Admin Portal, such as a flag in the database for personal hackathons, so that hackathons will not be sent to the Admin Portal if they are personal hackathons. This is an idea we contributed

during a design meeting with most of the people working on HackBox, and we are very happy that it is set to be added to the final product.

Chapter 6: CONCLUSION

The goal of this project was to develop a new administrator's portal, for Microsoft to use in their new version of HackBox, with the ability to manage the various hackathons, and their subsequent parts. Although the front end website to handle everyday interaction by users was being created by another team, admins needed a way to simple manage the hackathon data. The design for the Admin Portal is a modernized interface using blades, based on the Azure Dashboard design. The Admin Portal in conjunction with the rest of HackBox allows the Garage team at Microsoft to foster new ideas and projects that may benefit Microsoft and the software community down the road.

Our research showed that tools to manage a hackathon were few, and that there were many libraries and design ideas to help guide us in creating the Admin Portal. We implemented the web app using AngularJS, Node.js, and other frameworks and libraries. The Admin Portal successfully complements the front end website using the database developed by another team at Microsoft. The Admin Portal allows admins to manage their hackathons in a simple and clean interface that is more data oriented than the front end website.

Throughout this project we were presented with challenges and new problems to tackle. Through tackling these challenges, we gained skills in setting and reaching goals, along with agile methodology and project management. We also learned many new technologies from Node.js and AngularJS, to Sass and Chutzpah. At the start of this project none of us had much experience working with web technology such as JavaScript and Node.js, however now that we have completed this project we can consider ourselves well versed in front-end web work.

BIBLIOGRAPHY

i Vanessa Ho, “Windows 10 launches, //oneweek Hackathon exercises innovation muscles and Microsoft Imagine Cup winners announced – Weekend Reading: July 31 edition” (July 2015). [Online]. http://blogs.microsoft.com/blog/2015/07/31/windows-10-launches-oneweek-hackathon-exercises-innovation-muscles-and-microsoft-imagine-cup-winners-announced-weekend-reading-july-31-edition/

ii Ibid

iii Joshua Tauberer, “How to run a successful Hackathon”. [Online]. https://hackathon.guide/

iv Conversations with Steve

v Conversations with Steve

vi Hacker League. [Online]. https://www.hackerleague.org/https://www.hackerleague.org/

vii Hack Dash. [Online]. https://hackdash.org/https://hackdash.org/

viii Wolfram Development Platform. [Online]. http://www.wolfram.com/development-platform/http://www.wolfram.com/development-platform/

ix Microsoft Garage. [Online]. https://www.microsoft.com/en-us/garage/

[x] Andrew Connell, “The New Azure Management Portal Rocks!” (April 2014). [Online]. http://www.andrewconnell.com/blog/the-new-azure-management-portal-rocks

xi Node.js. [Online]. https://nodejs.org/en/

xii Chutzpah - http://mmanela.github.io/chutzpah/

xiii Git. [Online]. https://git-scm.com

xiv Github. [Online]. https://github.com

xv AngularJS. [Online]. https://AngularJS.org/

xvi JQuery. [Online]. https://jquery.com/

[xvii] “Angular Developer Guide /Scopes”. [Online]. https://docs.angularjs.org/guide/scope

xviii Azure. [Online]. https://azure.microsoft.com/en-us/

xix Javascript. [Online]. https://www.javascript.com/

xx Peter Wayner, “Here's how the old guard and upstart darling of the server-side Web stack up against each other” (January 2015). [Online]. http://www.infoworld.com/article/2866712/php/php-vs-node-js-an-epic-battle-for-developer-mind-share.html

xxi SASS reference. [Online]. http://Sass-lang.com/documentation/file.SASS_REFERENCE.html

xxii Less. [Online]. http://lesscss.org/

xxiii AD Graph REST. [Online]. https://msdn.microsoft.com/en-us/library/azure/hh974476.aspx

[xxiv] TinyMCE. [Online]. https://www.tinymce.com

xxv Leon Welicki, “Announcing Azure Portal general availability” (December 2015). [Online]. https://azure.microsoft.com/en-us/blog/announcing-azure-portal-general-availability/

xxvi Microsoft Press, “Active Directory”. [Online]. https://msdn.microsoft.com/en-us/library/bb742424.aspx