MYTRI: ENCOURAGING PARTICIPATION IN A PROJECT-BASED SOCIAL NETWORK

BY

VINIT SHAH

THESIS

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Electrical and Computer Engineering in the Graduate College of the University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Adviser:

Assistant Professor Shobha Vasudevan
ABSTRACT

Creating full websites to share projects can be difficult and time consuming. Finding members to work on said projects can be equally demanding. MyTri is a project-based social network that allows users to quickly share projects and find collaborators. Preliminary testing has shown MyTri’s project pages to be a viable option for sharing project information. This thesis will go over the development and design processes of building such a social networking website and provide suggestions for future work. More specifically, this document will provide examples that utilize visualizations and game-like features within a working social networking website. While these examples are useful, they are here to be used as stepping stones for future development of social networking websites (particularly when concerned with encouraging participation within these sites) rather than as a definitive guide.
To my family and friends, for their support and words of wisdom
ACKNOWLEDGMENTS

First, I would like to thank Professor Shobha Vasudevan. Without her, this thesis in any form would not exist. It was through discussion and deliberation along with her ability to adapt to new ideas that I have been able write this thesis. I would also like to thank Professor Karrie Karahalios, without whom I would not have understood the rich complexity and variety of research and literature in social spaces and visualization.

There were a number of people involved in the thesis who have affected its development in one way or another. I thank Neha Shehafizad, Linjia Chang, Guangzhe Gao, and Zhiyuan Jin for their contributions to outreach and early design development and express gratitude to the designers, Wenjun Wu and Miranda Singler, for their help in giving MyTri personality and depth.

Lastly, I thank my friends and family who have offered their wisdom and support throughout this process.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER 1 INTRODUCTION</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Goals</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Literature Research</td>
<td>2</td>
</tr>
<tr>
<td>CHAPTER 2 BACKGROUND</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Social Space Behavior and Usage</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Information Quality</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Visualization</td>
<td>8</td>
</tr>
<tr>
<td>2.4 Game Design</td>
<td>10</td>
</tr>
<tr>
<td>CHAPTER 3 IMPLEMENTATION</td>
<td>11</td>
</tr>
<tr>
<td>3.1 Software Development</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Designing Features: From Mentors to MyTri</td>
<td>23</td>
</tr>
<tr>
<td>3.3 Design Iterations</td>
<td>30</td>
</tr>
<tr>
<td>3.4 Testing and Results</td>
<td>45</td>
</tr>
<tr>
<td>CHAPTER 4 DISCUSSION</td>
<td>48</td>
</tr>
<tr>
<td>4.1 Challenges</td>
<td>48</td>
</tr>
<tr>
<td>4.2 Future Work</td>
<td>48</td>
</tr>
<tr>
<td>CHAPTER 5 CONCLUSION</td>
<td>50</td>
</tr>
<tr>
<td>APPENDIX A CODE OVERVIEW</td>
<td>51</td>
</tr>
<tr>
<td>A.1 Server</td>
<td>51</td>
</tr>
<tr>
<td>A.2 Front End</td>
<td>54</td>
</tr>
<tr>
<td>A.3 Build/Task Scripts</td>
<td>56</td>
</tr>
<tr>
<td>A.4 Package Management</td>
<td>57</td>
</tr>
<tr>
<td>A.5 Miscellaneous</td>
<td>58</td>
</tr>
<tr>
<td>APPENDIX B SURVEY</td>
<td>59</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>65</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

MyTri is an English adaptation of a Sanskrit word meaning “camaraderie.” Originally an endeavor to encourage female college students to pursue Science, Technology, Engineering, and Mathematics (STEM) fields, MyTri was meant to be a means of fostering this sense of camaraderie, so the name seemed apt and it stuck. This endeavor was intended to bridge social media and education in the form of a web platform. The first iteration of this idea involved building a service to help facilitate and promote interaction between undergraduate female students and female faculty or professionals.

Through research and design iterations, we were able to reveal marked flaws in the original conception. While the original motivation exists to some degree, MyTri has since evolved into social space for students to share projects they have worked on. The rest of this thesis will delve into the details of the challenges associated with developing such a system, where the design considerations came from, and what areas still need work.

1.1 Motivation

In 2009, the U.S. Department of Commerce released a report regarding the gender gap in employment in the United States [1]. A key point was the finding that women earn less money than men do —14% and 21% on average in STEM and non-STEM fields, respectively. While the employment rate was nearly evenly split between men and women for non-STEM fields, in STEM fields the divide is much larger, with women making up less than a quarter of the STEM workforce [1].

There is a clear need to bridge this gap, both in pay and in choice of employment field. Neither insight as to why these results exist nor mention methods for ameliorating the issue is mentioned by Breede, et al [1]. This
fact was part of the impetus for the development of MyTri. If women could be encouraged to participate in STEM fields at the university level, could social media, which are platforms used by increasingly many college students daily, be part of the solution in doing so? A number of efforts utilizing the web and social media have arisen to tackle the issue, each with a specific demographic or purpose [2], [3], [4].

In the process of designing the MyTri social network, the end result (which in itself is still a work in progress) evolved to fit a larger target demographic: college students as a whole as opposed to singling out only female students. It is not a panacea and needs much more rigorous testing outside of this controlled space in order to make broader conclusions. However, it serves as a framework for a model utilizing technology and social media in order to make positive change.

1.2 Goals

It was useful when starting the development of MyTri to come up with goals for the end product. As mentioned earlier, MyTri’s original goal was to create a digital social space capable of encouraging female college-level students to join STEM fields.

In order to narrow this down into something that could be built, we identified three goals:

1. The space should be *useful*.
2. The space should encourage social participation.
3. The space should encourage repeated use.

Chapter 3 will detail each of these goals and how they have evolved.

1.3 Literature Research

There tends to be much literature on the effects of social media after the fact [5], [6], [7], but there is little on the actual process of development as a whole: this includes both feature design and testing as well as technological development.
The interest of the literature research was to understand how people act in the context of social networking websites and what motivates them to do so. While we understood the literature could not tell us everything about the success of social media, it would provide us with a good basis to determine whether or not MyTri would be feasible.
CHAPTER 2
BACKGROUND

Much of the conceptualization of the current website, both in terms of visual design and features, was inspired by or adapted from existing literature in the broad fields of psychology, human computer interaction (HCI), and visualization. Users will find similarities to existing technologies as well. These include social features from platforms such as Facebook and LinkedIn, for example, but even these features are explored in the literature [6], [5]. One of the primary concerns for delving deeper into these areas was to understand how behaviors and features could be expanded upon in a way that would motivate potential users.

Each section is a research area in and of itself, but it is not treated as an exhaustive survey of literature in the area. Rather it is provided here as an overview of some of the necessary information that helped shape the design of MyTri.

2.1 Social Space Behavior and Usage

2.1.1 Signaling

A 1993 study looked at how behavior in groups differed when conversations happened in person rather than online [8]. It found that online interactions (particularly those that were not face-to-face) tended to be more democratic than their in-person counterparts. Online discussions had less of a tendency to favor higher status individuals. For example, a master's degree student in a group consisting of mostly undergraduates would not hold as much sway online (face-to-face members would generally agree with the master's degree student). A consequence was a polarizing of opinions; online discussions tended to converge less quickly and lead to more heated arguments.
Preventing these sorts of exchanges seems pertinent in a time when anonymous or pseudonymous discussions can take place on many popular websites.

In order to better understand these types of interactions and the motivations behind them, a framework of understanding was necessary. Signaling theory is a field in both economics and biology that deals with communication between individuals [9], [10]. Donath [9] uses signaling theory to describe features of Social Networking Sites (SNSs). She argues that by utilizing elements from signaling theory, SNSs can be used as more effective “social tools.” This thinking is in line with the first two objectives of MyTri: the usefulness of the platform (a tool) and the encouragement of social participation.

Signaling information such as the authenticity of a person’s identity is one such example of where signaling theory and SNS behavior intersect. Donath [9] uses LinkedIn, an SNS which focuses on one’s professional network, as an example. On LinkedIn, if Alice wants to connect with Bill, she must be able to input his email address. While not completely foolproof, it does help prevent connections between unacquainted individuals. Here, LinkedIn has effectively strengthened the signal of making a connection, because a third party familiar with LinkedIn would be aware that, at the very least, Alice had Bill’s email.

The trade-off between signal cost and strength plays an important role in developing the rationale as to why certain features should and should not be incorporated into MyTri’s design. Signaling theory as a whole provides context for such a discussion.

2.1.2 Social Network Usage

Recent literature in the area of social spaces has targeted Facebook as a space ripe for study, and with good reason. With ten years of operation and hundreds of millions of users [11], Facebook is a sturdy platform with a history rich in social interaction [5].

It is useful to understand that Facebook use has not changed significantly over the years (from 2006 to 2008), even with its interface and feature updates [5]. Lamp et al. conclude that usage was mainly in maintaining “lightweight
contact with relationships... developed offline.” Such a finding is consistent with [12] which predicted that technologies would help facilitate weak ties as opposed to strong ones. Weak ties refer to relationships between individuals who are acquainted in a limited context (such as a colleague, classmate, or co-worker) and strong ties are those with whom an individual has frequent contact with multiple foci, or areas of interest (such as a family member or mentor) [12]. Donath and Boyd [12] argue that weak ties expose an individual to new information and opportunities.

Opportunities do not explain all of the features on Facebook (or other SNSs, for that matter). Xho et al. [6] argue that usage on Facebook and other SNSs can be grouped into one of three broad categories: performance, exhibition, and personal archive. They argue that people use social media for multiple purposes and that “designs should respect that.”

The public actions that a person takes, such as uploading a photo or submitting a status update, fall in the realm of “performance.” The tagging of many friends in a photo or publicly wishing a friend “Happy Birthday” on her Timeline, for example, indicate social capital and grooming, respectively [9]. These performances are some of the key features of Facebook as they provide its content, especially that which is recent. This appears to be the crux of the paper’s argument, that is, spatial and temporal regions can create tensions. For example, an item in the performance region (something which is recent) can later be archived (made private). Xho et al. [6] argue that systems should use designs that promote saliency in what is public and what is not; a feature that users mentioned Facebook lacks. A user mentioned making an image private to herself, but was confused as to how to view just the ones that were private [6].

Metaphors appear to be another area of concern. Facebook employs many metaphors, such as the Timeline and the News Feed. Xho et al. [6] stress that metaphors can play a large role in how a website is used. The News Feed, for example, helps with the performance region, because with it, the cost to see content created by friends is decreased, though the strength of the signals they send does not necessarily suffer.

SNS usage is not, of course, limited to the above use cases and there are many other SNSs that could potentially provide insights in different areas. What we found important for our purpose, though, were the findings that discuss design considerations, as these can be broadly applied outside the
context of a particular SNS. Determining which metaphor to use when displaying old data is clearly important. For example, the question of badges and tags (discussed further in Chapter 3), were metaphors that would determine in what lens a user viewed the particular metadata of a project.

2.2 Information Quality

Apart from understanding the behavior of SNS usage, it is also useful to understand how metadata, such as the popularity of a song, for example, can have an effect on the user experience of a system. Users may tend to “herd” around songs at the top or bottom of a popularity chart and add to its popularity (a “herding” behavior) [13]. If the organization of data has such a large effect on the way it is received (in this case, how many downloads a song gets), it is necessary to understand why.

Users tend to herd around that which is described as being high quality [13], at least to some degree (the existence of self-fulfilling prophecies outside of controlled tests is still debated). It is the responsibility of a system designer, then, to make sure that the metadata associated with a piece of information (such as a song) reflects its actual qualities in systems that order items by quality. Another example of this would be search engine results, in which a user would probably want to see the most relevant items at the top of a list. Of course, the problem in quantifying quality is no easy task and is dependent on the item. It is not the purpose of this thesis to define “quality” or the proper methods for ordering items, but rather to take a look at solutions some popular systems employ in order to gain an understanding of how to better design the MyTri website.

In systems where the quality of an item is determined by user feedback, there is always the possibility of underprovisioning where an item gets so little attention by a user that it goes largely unnoticed [14], [15]. Consensus on the cause of underprovisioning or its remedy has not yet been reached, but there do appear to be some design features to take into consideration. For example, on Reddit, a popular website composed almost entirely of links to other web pages created and voted on by users, most content (something on the order of 90-95%) goes largely unnoticed [14]. Here, the design itself might be the issue. If 95% of content will not be worth someone’s time, it may be easier
to just wait for the best content to bubble up instead of weeding through all posts to find the 5% worth viewing. Gilber [14] also suggests that perhaps a user needs to see a post multiple times before deciding to finally vote on or view it. On Facebook, this is a feature rather than a large issue. Cascading is a case where users re-share content they have seen from someone, potentially in a different network, into members of their own network [7]. Dow et al. [7] found that users tend to share (or re-share) content seen from popular Facebook pages. For example, when a celebrity shares a picture, many users tended to re-share from those pages (the celebrity’s large network probably plays a large role in this). This finding is, to some degree, consistent with [15] in that posts from users with high status or value tended to receive the most attention.

In order to create an SNS that is meant to allow for students’ projects to be displayed and searched for, developing an interface that allows each project to fairly gain recognition is pertinent. It may not be feasible, for example, to have a series of trusted moderators in the same way that Slashdot rates and ranks content; and even these systems are shown to be fallible [15]. In this regard, the solutions a design employs will probably need to be tailored to the features of the particular SNS. As of yet, there does not seem to be a silver bullet for quantifying quality and avoiding underprovision.

2.3 Visualization

When one thinks of Facebook or Twitter, “information” and “visualization” are probably not the first words that come to mind. In the design of MyTri, though, visualizations play a crucial role. They serve as a means of summarizing data about people and projects and do so in a way that is succinct, informative, and visually appealing.

It is important, therefore, to understand how to convey meaning appropriately via a visualization. Hullman and Diakopoulos [16] describe editorial layers, namely the data, visualization representation, textual annotations, and interactivity, as places where meaning can be conveyed. Each layer can be manipulated in order to achieve some effect. Omitting certain data points, for example, may be beneficial in order for a visualization to maintain simplicity [16].
Understanding how these techniques can lead to bias is important, especially if the biases can be unintended. Anchoring techniques, that is, techniques which direct a user’s attention such as providing a default view of a visualization, can frame a user’s perspective to take into account the default view as the most important [16]. A visualization that charts societal features on a map, for example, may filter, by default a characterization such as race. Doing so can have the unintended (or intended) consequence of making racial issues the main message of the visualization due to its salience.

Freeman [17] describes a history of network diagram visualizations used to denote social connections. In the article, Jacob L. Moreno is lauded for his early depictions of social networks in graph form, particularly his use of arrows (for direction), colors for multigraphs, varied use of shapes, and variation in locations of points to stress structural features [17]. Today, visualizations liked LinkedIn’s Maps [18] utilize some of those original features, but also add a level of interactivity by allowing users to click on vertices to get a more detailed view of an individual in one’s network.

While LinkedIn’s network visualizations feature data from a professional network, it likely falls under the definition of a Casual Information Visualization (Casual InfoVis) by Pousman et al. which depict “personally meaningful information in visual ways... in both everyday work and non-work situations” [19]. MyTri’s InfoVis should fall under this category as well; the visualizations MyTri provides are not necessarily made available for rigorous data analysis, but rather are meant as a means to summarize (or reflect on) project details and motivate further usage of the website. A more apt example would be the Wordle Casual InfoVis, where users could input large blocks of text and a program would output a text-based visualization [20]. We say that Wordle is a more apt comparison to MyTri’s project visualization because of its creativity aspect. In a Wordle study, 70% of users describes feelings of creativity when authoring and sharing Wordles [20]. What this tells us is that the mere act of creating a visualization, even if the creation process requires minimal effort, can be intrinsically rewarding for an individual. Aside from Wordle’s appealing design (colors and typography were appealing for users), Viegas et al. [20] offer that this positive reaction may be due to the fact that a user’s Wordle has the potential to look unique, especially with reference to one’s own social network (many users would share their Wordles with their friends).
2.4 Game Design

When describing the MyTri development process, it is necessary to discuss game design. Much of the influence of the current iteration of MyTri’s features comes from ideas in game design. In particular, the area known as “gamification” has been rising in popularity with businesses (including SNSs) [21]. Literature on gamification appears to be sparse and disjoint, so understanding the consensus on this field is difficult [22]. Deterding et al. [22] provide a definition that says gamification is “the use of game design elements in non-game contexts.” For the purposes of MyTri development, we have adopted the same definition.

There are many success stories of SNSs that utilize game principles such as FourSquare for location-based updates [23] and SuperBetter for self improvement [24], [25]. What makes these sites gamified are the game elements that they utilize. In order to develop a system that motivates usage in the way games do, it is useful to look at what elements of game design are available for usage. Werbach and Hunter [21] mention the Points, Badges, and Leaderboards (PBLs) as the “skin-deep” knowledge of gamification. Here, “points” refer to a score users might be able to increase with proper gameplay (or website usage). Badges are virtual awards earned for an accomplishment—KahnAcademy [26] and StackOverflow [27] are popular examples of sites that utilize such features. Leaderboards refer to rankings of individuals against a community best on their performance.

Werbach and Hunter [21] define PBLs as components in a larger set of game elements, which are themselves places in one of three categories: dynamics, mechanics, and components. It is not necessary to get into the details of these components, but it is useful to understand where PBLs falter and how game elements can be integrated into website design without disturbing the site’s main purpose.
3.1 Software Development

Because MyTri is a software-based tool, much of the time spent to develop MyTri was devoted to the software aspects. There is considerable literature on the design of SNSs, but little on how to organize its software. It is pertinent, then, to have a log of the major software design decisions made, with particular emphasis on how and why those decisions were reached, so that future developers can look at this section as a reference point for continuing further development of this project.

MyTri’s development philosophy tends to be consistent with the trending Agile methodology [28], in that the focus on development was on small iterations. Given that MyTri’s development also required a thesis component, it was difficult to completely adhere to the Agile philosophy, as the decisions made in the development of MyTri needed to be well-informed and carefully considered, coded, and then finally executed.

3.1.1 Software Stack

Choosing is a software stack can be a tedious and confusing process, especially when it involves weighing the many different options available. It is necessary then, to decide a set of requirements to serve as a baseline for choosing a technology to use. In the case of MyTri, the priority was rapid development so as to gain quick feedback on its viability as a useful tool. Further, there were considerations about performance and scalability to weigh in as well. This section is not meant to be a de-facto standard for choosing a stack for SNSs. Rather we provide a set of choices and arguments for making those choices as well as other potential options should the need arise to change a
Table 3.1: Server Stack

<table>
<thead>
<tr>
<th>Web Server</th>
<th>Node.js</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistent Storage</td>
<td>MySQL</td>
</tr>
<tr>
<td>Session Storage</td>
<td>Redis</td>
</tr>
<tr>
<td>Operating System</td>
<td>Ubuntu</td>
</tr>
<tr>
<td>Programming Language</td>
<td>Javascript</td>
</tr>
</tbody>
</table>

technology medium.

LAMP (Linux, Apache, MySQL, PHP) stacks are popular choices for rapid web development, as websites tend to “just work” out of the box. For MyTri’s development, we decided to forgo using an Apache server and the PHP language in favor of nginx (pronounced “engine x”) and the relatively new Node.js framework while hosting the server on a virtual Ubuntu machine and storing the data in MySQL. Table 3.1 shows the stack used for MyTri’s development.

Node.js, Javascript

The choice of using Node.js (Node), as with any programming language or framework, was probably the most important, as it is the language in which much of the server-side code would be written. Node.js is a runtime environment and library that allows programs to be written in Javascript [29]. Its main point of attraction is its event-based model for I/O as opposed to the thread-based model used in servers like Apache for handling multiple connections. Node developers claim that this provides for better scaling when dealing with many simultaneously connections, as spawning new threads and processing (and then performing context switches between them) can be expensive on the CPU and memory. Nginx [30], a proxy server-like Apache, serves requests via asynchronous events, which is ideal for serving static assets and forwarding requests to the asynchronous Node.js server.

While the potential performance boosts are useful, the decision to choose Node.js as a framework relied more on the programming conventions associated with it. Since Node.js programs are generally written in Javascript, only one language is necessary between the client-side (browser) and server-side as Javascript is natively supported in all modern browsers. Moreover, the single-threaded model in Node.js allows for programs to be written without
dealing with synchronization issues between connections.

The trade-off in choosing Node.js as a framework is that it is a new framework using an event-based, asynchronous programming paradigm, which is not necessarily conventional in web development. Being a new framework, Node.js does suffer from API upgrades and stability issues, though during this development cycle, there were no major breaking changes to the API that affected MyTri’s codebase.

MySQL and Data Storage

The rest of the decisions regarding tool choice were more for simplicity and developer experience. MySQL [31], for example, is a widely used relational database, so finding documentation and examples in getting a development environment started was relatively easy to do. Moreover, the University of Illinois provides MySQL databases free of charge for student-run projects, allowing for ease of hosting through university support.

It is important to note the choice of using a relational database as opposed to the many other options available for data storage and management, such as MongoDB, couchDB, etc. While solutions such as MongoDB [32] provide an object-oriented (or document-oriented) model, mapping some of the relational elements to a document-oriented database could prove cumbersome. Much of the data in MyTri, such as tags, users, posts, and projects, have many-to-many relationships and utilize relational joins and aggregate functions. While it may have been possible (and still remains a possibility) to use a document-oriented approach, it seemed more pragmatic to stick with the choice that was familiar and could still handle the data.

Finally, the choice of using Redis [33] as a key-value store to hold user session data was primarily due to its simple integration with the Express [34] framework via the connect-redis module [35].

3.1.2 Frameworks and Tools

Much of the MyTri code utilizes outside frameworks as a means for simplifying the code base. In many cases, like in choosing the back-end stack, there were many options available. This section is not intended as a full argument for or against any particular framework, but rather serves as a means for
future MyTri developers to understand what frameworks, tools, and libraries are used in the code and why.

Server-Side Frameworks: Express Routing, Jade Templates, and Other Modules

Subsection 3.1.1 mentioned the use of the Express framework for Node.js [34]. Express is primarily used as a means to make it easy to develop Node.js webservers.

```javascript
var http = require("http");

http.createServer(function(req, res) {
  /** Home Page **/
  if (req.url == "/") {
    res.writeHead(200, { "Content-Type": "text/html" });
    res.end(/** Home page response **/);
  }
  /** Second Page **/
  else if (req.url === "/api/secretpage") {
    if(condition === met)
      res.end(/** Welcome to the secret page **/);
    else
      res.end(/** Condition not met **/);
  }
  ... Third, fourth, fifth, etc. routes
}).listen(80, "localhost");
```

Figure 3.1: Node.js Vanilla Routing

Figure 3.1 displays a method for handling routes that can get much more tedious when many API routes are used or when there are many conditions that need to be met before a response can be sent.

Express uses a more concise and robust mechanism for handling different routes. It parses route names using regular expressions and allows for “middleware” methods.

Figure 3.2 provides the same functionality as Figure 3.1, but as can be seen has much more reusability in terms of the “conditionHandler” middleware and “*” wildcard expression for the routes beginning with the string “/api/”.

14
function conditionHandler(request, response, next) {
    /** if the requester has met our condition:
        e.g. The user is logged in **/
    if (request.condition = met) {
        /** Then continue on to the next method **/
        next();
    }
    /** Otherwise send back a 403 response **/
    else {
        response.statusCode=403;
        response.send('Condition not met');
    }
}

/** Require that all routes starting with '/api/'
 meet the condition **/
app.get('/api/*', conditionHandler);

app.get('/api/secretpage', function(request, response) {
    response.send('/** Welcome to the secret page **/');
});

Figure 3.2: Express.js Routing

route('post',
    '/api/project/:id/remove/member',
    [minAccess(`ADMIN`), ProjectHandler.removeMember]);

Figure 3.3: Advanced Routing and Middleware

Using some syntactic sugar (the “route” method) and middleware concepts, MyTri uses Express to make routes relatively easy to read and thus debug. Figure 3.3 shows a line in which middleware is used to make sure only an administrator of a particular project can remove project members via the API.

Express allows for easier route handling, but it is only one of many different modules (libraries in Node.js) used in the MyTri source code. Modules like connect-redis [35] and node-mysql [36] offer Node.js connectors to Redis and MySQL, respectively, while libraries like async [37] make it much easier to deal with Node.js’s asynchronous nature. It is outside the scope of this thesis.
to go into further detail about all these modules because they generally serve very specific purposes and could be switched out with other modules that provide similar functionality.

It is important to note the use of server-side templates. Jade is the templating engine used in MyTri’s back-end [38]. The decision to use a templating engine at all (as opposed to using a particular engine like Jade) is what is important. Templates refer to segments of front-end code (in this case HTML) that can also take variables as inputs. This was important, because not all users will see the same thing.

```html
    title=user.name + "\'s profile page"
```

Figure 3.4: Simple Jade Example

A template in Jade that looks like Figure 3.4, might render “Alice’s profile page” for one user and “Bob’s profile page” for another. What the templating on the back-end allows is for flexibility with code and a separation of concerns. Code related what a user sees is located in separate “.jade” (or any templating extension) files which take their own set of inputs.

```text
    On request:
    Determine route
    Fetch form or write to databases
    Render page information
    Send back data or page information
```

Figure 3.5: Psuedo Code for Handling an HTTP Request

Figure 3.5 shows a high-level view of the steps taken for handling an HTTP request. Each one describes a different separation of concern and, in the source, generally a different set of classes and modules which handle the concern. While it is not necessarily pertinent to know the type of templating engine to use, it is necessary to understand the split in the source code consisting of templates, route handlers, and database controllers.

**Client-Side Frameworks: jQuery, Backbone, Bootstrap, and d3**

Much of the work in developing the front-end was made easier with the use of a variety of frameworks, each with its own purpose. This section will go
jQuery [39] allows for easy access to the HTML Document Object Model (DOM) elements via Javascript code. Figure 3.6 allows a single line of code, i.e. \$\("form\).submit\(\)\, to act in the place of building a loop or iterator set these elements one by one. jQuery’s ability to abstract away many DOM elements and native Javascript functions was the primary reason to choose this framework.

Making the website mobile-friendly was one of the early design challenges. Given the increase in phones and tablets with browser capabilities, it seemed natural to at least offer web pages that could be viewed in smaller-sized screens. It was not feasible to create an entirely different web page or app given the time constraints of this thesis, but by employing responsive web development and design principles via the Bootstrap framework, the MyTri pages were able to fit into smaller browsers without loss of content or information [40], [41].

Twitter’s Bootstrap frameworks provides a set of Javascript libraries and
CSS classes that abstract away much of the boilerplate code associated with creating web pages that respond to a browser’s height and width.

The example code snippet in Figure 3.7 lets the browser’s renderer handle what to do in the case that its size is “medium”, “small”, or “extra small.” Bootstrap uses a 12-column system, so a class like “col-xs-3” requires that in the case that the browser’s width is “extra small”, the width of the current element should not exceed three columns. Figure 3.8 shows what the browser might look like in the case that the browser is large, or standard desktop size, and extra small, standard mobile size, respectively.

jQuery and Bootstrap handle much of the interface layout portion of the client-side code, but in order to keep a separation of concerns on the front-end, the Backbone.js [42] was used. Backbone provides the View, Model, and Collection helper classes in order to keep code sections distinct. With Backbone, MyTri was able to follow a Model-View paradigm on the front-end. With Backbone, we were able to use a single collection of objects—“tags” for instance—and use them in different parts of user interface (UI). Figure 3.8a shows tags that are visualized as a series of circles and that are also rendered with their names and amounts displayed further below. Both of these representations use the same underlying data, so changing the data will effect all parts of the web page that use tags. It was for this ubiquity and decoupling of data and interface that Backbone was chosen as a framework.

Data Driven Documents (or D3) was the final major framework utilized on the front-end. D3 is a Javascript library, similar to jQuery, in that it allows easy manipulation of HTML visual elements [43]. Figure 3.9 describes
Figure 3.9: Visualization Update

```
<svg>

<div id="tag-{{id}}" class='col-xs-6 col-sm-4 col-md-3 tag'>
  <div class="tagBorder">
    {{tagCount}}
  </div>
  <p class="tagName">{{name}}</p>
</div>
```

Figure 3.10: Simple Handlebars Example

D3’s update pattern which lets data be visualized via HTML’s `<svg>` element without using a page refresh. The example shows the top three tag visualizations being updated once an “EPA” tag is removed.

It is also useful to note that the client-side uses its own templating engine called Handlebars [44] so as to make certain segments of HTML code reusable. Like Jade on the server-side, Handlebars allows pieces of HTML code to be created using variables. Figure 3.10 shows the template code for the tags shown in Figure 3.8. Everything is standard HTML aside from the text wrapped inside “{{ }}” which are how Handlebars represents variables. Handlebars provides the ability to precompile templates to speed up execution time. Handlebars’s precompilation also works well with MyTrí’s build tools and it is in the build-phase, prior to starting the server, that the compilation occurs. With the modular design of the code, it is possible to switch to other templating engines in the future.
Table 3.2: Development Tools

<table>
<thead>
<tr>
<th>Version Control</th>
<th>Git</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build</td>
<td>Grunt</td>
</tr>
<tr>
<td>Client-Side Package Management</td>
<td>Bower</td>
</tr>
<tr>
<td>Server-Side Package Management</td>
<td>npm</td>
</tr>
</tbody>
</table>

Managing Code

To make the process of managing MyTri’s codebase easier, we have employed a few useful tools for development. Table 3.2 provides an overview of the tools used and their respective purposes. Git [45] was the version control system used as it makes it easy to deploy production code to other platforms. Grunt [46] is used to make scripts in Javascript. MyTri uses these scripts during development and when pushing to production. A call to “grunt build:watch”, for example, lets front-end code be compiled as soon as changes are made to simulate what it would look like in a production environment. Bower and the node package manager keep track of libraries and their versions, so an execution of the “bower install” and “npm install” packages will install the latest versions of client-side and server-side libraries, respectively.

3.1.3 Software Design

Relational Schema

The relational schema for the MyTri website went over a few iterations during the process of determining what its features would be. Figure 3.11 shows the latest iteration of the database schema. The diagram omits most of the table attributes for visibility purposes. The “parties” table was used as a way to create unique IDs between tables. Since users, projects, and posts share similar features, assigning a unique ID across all of those tables seemed necessary in order to prevent table “explosion” and nullable attributes. By explosion, we refer to the number of “tableA_has_tableB” tables (such as user-HasTag, postHasTag and projectHasTag). The other option is to have just a “hasTag” table, but attributes such as “user_id, project_id, and post_id” where just one of the attributes is set. It seemed slightly more pragmatic just to have a single “owner_id” whose foreign key is indexed in the “parties”
Aside from the party model, the rest of the schema is built around what features the website would support: for example, tags and skills having “hastags” and “hasskills,” respectively. The reason the hastags table also associates with a “tag” table is because it was assumed that tags would be more ubiquitous between projects and thus shared and used as search terms. Skills, on the other hand, are particular to each project and project member, so it did not seem necessary to create an entirely separate “skills” table. If usage dictates otherwise, though, it might be prudent to add a skills table and an index on a skill ID.

Separation of Concerns

Apart from the relational schema, which is handled by the database, it was necessary for the rest of the code structure to stay organized in order to reduce bugs and unreadable code. The “Model-View-Controller” article [47] (based on an earlier discussion of SmallTalk [48]) describes a methodology for separating code into three distinct modules: the Model, View and Controller (MVC). MyTri utilizes the Backbone.js [42] framework in order to achieve a similar separation of code, but in this case there are just models and views. The models handle data retrieved from the server and views handle render-
var TagView = Backbone.View.extend({
    render: function(){
        var renderedHTML =
            this.handlebarTemplate(this.model);

        this.element.html(renderedHTML);
    }
});

Figure 3.12: Tag View Rendered Using Model

ing that data. User input—the “controller” portion of MVC—is handled generally by either views or helper methods and objects.

The models are represented as “Model” class objects (or “Collections” of these objects). These interface with the objects in the “View” class, e.g. a “Tag” view would render its Handlebars template with a tag model. Figure 3.12 describes a simplified example of the TagView rendering its template using its model. The relationship between models and views tends to be either 1 to 1 or 1 to many, in that multiple views can reference the same model as in Figure 3.9.

While this approach is not a strict MVC one, the separation of concerns that it does utilize provides enough distinction to allow for code reuse and better readability without being overly complex.

3.1.4 Privacy and Security Considerations

As a website that handles and stores sensitive user information such as passwords and email addresses, it is important to discuss some of the precautions taken in MyTri to prevent such data from being compromised.

For password storage, MyTri utilizes a node.js implementation of the bcrypt algorithm for password storage [49]. The algorithm ensures that passwords will not be stored in plaintext using a salted hashing algorithm. This precaution prevents attacks from rainbow tables. Bcrypt’s work factor suppresses brute force attack times by slowing down the hash function.

Outside of password storage, node-mysql [36] escapes characters using prepared statement-like syntax. The purpose of escaping SQL characters in this
manner is to prevent SQL code from being injected into a query and allowing malicious users to gain access to potentially secure data. Future work on this project might look at non-SQL databases, which generally do not suffer from SQL injections, since they do not use SQL.

All inputs also go through a sanitation process to prevent Cross-Site Scripting (XSS) attacks using a version of Google’s Caja sanitizer [50]. The sanitation process converts all inputs (including null and undefined values) to strings and trims whitespace on ends.

Future work will look at the usage of OAuth [51] to login to MyTri via a third party such as Google. This will prevent the necessity of password storage altogether.

3.2 Designing Features: From Mentors to MyTri

As with any user facing product, much of the emphasis of MyTri was to determine what features should be implemented and, more importantly, how they would affect use of the website. This section is based primarily on literature and product research. Section 3.4 will discuss the feedback received and how MyTri might move forward.

3.2.1 Shift from Mentor-Based Network

The original plan for the development of MyTri was to build a mentoring system for young female students to be mentored by the more experienced women in the STEM industries. What would set MyTri apart would be its use of social media to streamline the experience. Our first few discussions and design iterations were in line with this idea, but after more research we found that many others had already implemented similar systems. One of the most popular was MentorNet, a mentoring program that connects any student in a STEM field with a STEM professional [52]. While MentorNet is available to students of any gender, it initially focused on closing the gender gap [1] by connecting female STEM students with female professionals.

Apart from the influx of mentoring SNSs, which generally connect females already in STEM fields, MyTri’s discussions became more in line with sites like Passion Projects [2]. With this approach, the idea is to inspire women to
join STEM fields by displaying the sorts of projects undertaken by people in STEM fields; in the case of [2], these projects are primarily software related. Drawing upon these ideas, MyTri’s designs started focusing on the idea of displaying projects to encourage interaction and collaboration.

3.2.2 Encouraging Interaction

Encouraging interaction, both with the website and with other users, is important in any SNS or tool. With MyTri, it is especially important, as the website’s focus is intended as an aid for students to collaborate with each other. There are many real-world examples of effective platforms that allow for seamless user experience and collaborative interaction. Facebook’s Messenger application, for example, allows multiple users to engage in the same conversation thread. Wordle, while less socially aware, was an effective creativity outlet, particularly because it allowed users to share their work with an audience [20].

Badges and Gamification

Points, badges, and leaderboards work well for game-oriented social spaces such as FourSquare [23] and were in some of MyTri’s earlier designs. Without a proper narrative or reason for using these game elements within MyTri’s design, these ideas seemed to be more of a distraction than an encouragement for participation in the website. Similar ideas are mentioned in [21].

At the heart of these features is denoting status or social capital within a community. StackOverflow provides working examples with its reputation points and badges shown in Figure 3.13. Here, reputation points are
earned from the community by answering questions correctly (see Figure 3.14). Rather than have the system itself give the rewards, users, instead, give the rewards. Many other question-answer based websites use similar systems [15], [14].

### 3.2.3 Tagging

The idea that the gamification aspects would be controlled and monitored by a community is where MyTri’s tagging feature drew inspiration. “Tags” in MyTri are short, descriptive pieces of text which describe a project. When users search for projects on the website, the database looks through the “Tags” table to find any matches. Instead of using points and badges to represent abstract ideas, we decided to let tags serve as the gamification mechanism. In this regard, tags serve three purposes. The first, already mentioned, is that they are descriptors for projects which allow for easier searching. The second is that tags allow for users to send low-cost signals to a project’s members of their interest in the project. Last, these signals are aggregated to form a pseudo-reputation system for a project.

---

Figure 3.15: Tags in MyTri

Figure 3.15 displays how multiple users can add the same tag to a project page so that it acts as a points system. The subheader text describes a tag’s purpose, so it is clear how tags have actual meaning within a real-world context (as opposed to solely within the narrative of the website).
Skills and Roles

Aside from tagging, which is a mechanism anyone can take part in, project members can add in “skills” to a project. These skills are pieces of text, similar to what one might list on the skills section of a resume. The reasoning behind putting these here, alongside tags, is that they describe a project in terms of its members—a skill is what a member is proficient at doing. Figure 3.16 shows an early mockup of skills on the website.

Roles are similar to skills in that they describe a member in relation to a project. Figure 3.17 shows how John Doe has taken on the role of senior engineer. This seemed like a natural feature to put within a project page since
it aids in establishing a user’s identity within the project group. Moreover, Figure 3.18 describes how a project can notify others that it is still looking for members. If a user is able to mark her own role within a project, it seemed like a logical next step to also mention if a project were in need of others to fill in available roles.

![Figure 3.19: Links to Members’ Profile Pages](image)

Badges serve as visual representations of some achievement or work in most gamification systems. They provide the “feedback loop” that keep users motivated to continue taking part in the system [25]. The latest design of MyTri does away with “traditional” badges, and instead opts to utilizing visualizations as literal visual representations of a project’s tag count. Figure 3.19 shows the section of the project page with visualizations. These visualizations update as data changes—the number and types of tags and skills. Because these update as data becomes available, the visualizations are meant to provide the same feedback loop.

![Figure 3.20: 'Traditional’ Badges in KahnAcademy](image)

Further user tests should hopefully provide insight into the efficacy of these
features as actual motivating tools. The option to employ more traditional gamification techniques is still available, should we find they are necessary.

### 3.2.4 Navigation and Discovery

In the paper on social translucence [53], Erickson and Kellogg argue how visibility of others within a community is important. The task of utilizing “translucency” as opposed to “transparency” was somewhat challenging.

Erickson and Kellogg [53] describe this difference as transparency allowing everyone to see all facets of everyone else in a community. Transparency, on the other hand, only allows for the visibility of a few items in great detail and the rest in some detail. From a design perspective, this makes sense. They argue that too much information at once can be overwhelming, but too little information may result in users who are not interested in learning about other ideas. MyTri’s attempt at social translucence was to have a “featured projects” section and a “featured students” section as some of the first pieces of information a user sees. Figure 3.34 in Section 3.3.2 contains a mockup of this design.

This provides users with a starting point to search for projects by minimizing the overwhelming prospect of looking for projects without an idea of what is available. The featured projects section looks at the top three projects with the most tags and then features them alongside two of the most recently created projects. Better methods of determining what should be featured is part of the future work of MyTri, since the current method can potentially lead to severely underprovisioning what may be perfectly good projects [14].

Navigating through different projects was meant to be part of the process of interacting with MyTri, so it was necessary to be able to get from one project to another, or from one user to another, relatively simply.

Each project has a “collaborators” section where users can see who is working on a project. This section provides links to each user’s respective profile page. A profile page has a list of the projects a user has worked on, so it becomes relatively simple to navigate from a project to a user and back to a project.
3.2.5 Summarizing Information

An important feature of MyTri is that it would allow for users to spend minimal time creating a web page, but still have that web page be representative of a project and its goals. While we realize that creating a fully developed website for a project has its advantages, particularly in fine-grained control of its design, when looking through the project websites of many professors and graduate students, we found many to be either overly verbose or succinct to the point where it was not clear whether the group was looking for new members.

Figure 3.21: A Wordle Example from the Wikimedia Commons Library

Visualizations in MyTri, mentioned earlier as allowing for a feedback loop, are meant to aid in the summarization of a project. More research needs to be done to see if tags and skills are the right data points to be visualized, but the literature seems to show, at least, that InfoVis’s are appealing to view and are most probably here to stay [19]. Figure 3.21 shows one such visualization in the form of a Wordle. MyTri attempts to emulate, to some degree, Wordle’s ability to take simple pieces of text and output a much more visually appealing creative work [20]. Viegas et al. [20] discuss Wordle as “participatory visualization”, and reported on how most users felt creative when making Wordles, even though the creation process is a matter of copying and pasting text. They say that Wordle was effective, in part, because it is interpreted less as a viewing tool and more as an authoring tool. While MyTri does not necessarily provide the same level of non-linear visualization behavior as Wordle, it is designed to draw on some of these aspects with its use of font size and updating visualization, using plain text as the underlying
3.3 Design Iterations

This section will discuss some of the major iterations in MyTri design and why changes were made. Most images will be mockups since many of them did not make it into the implementation phase. Many of the iteration changes were informed by discussion and mock usage. True understanding of whether a design “works” comes from testing it with users, but as a starting point, we used the knowledge we had available in order to iterate over the designs.

The following sections will contain only the most representative mockups and stills of a particular design idea.

3.3.1 Wireframes

Originally, we thought to make discussion the primary focus of MyTri. The idea was to bring in discussions happening from other websites, such as Twitter and display them on the MyTri website as part of the home page. Figure 3.22 shows an early mockup of this idea. This ended up being rather confusing to look at as well as to understand its purpose. Social media integration might still be in MyTri’s future, but perhaps worked more closely in with MyTri’s goals.

Before MyTri became focused on student-run projects, the idea was to
allow for any group of people to use MyTri. Figure 3.23 displays the original wireframe for an example group’s page. Note that the current page neither allows for a summary of what the group is or does, nor does it offer awareness of other groups or people aside from the discussion and members section. As mentioned in Section 3.2.4, social translucence is an important component in MyTri’s design. Further iterations in the design would take more ideas into account from [53] which make people and projects more visible.

Figure 3.24 shows what a MyTri profile page might have looked like. Again, it is important to note how the focus was on discussion, so nearly a third of the page displays discussions a user may have had. The yellow, gray, and brown items underneath the user’s picture were the original conception of badges in MyTri. Here, a user might be awarded a badge for excellence in the use of a particular programming language. The idea of using roles when displaying information about a group was brainstormed at this time as well. With this iteration, the roles were pieces of text that were written by other users, presumably group members. This concept was later split into the tagging and role features that exist now for clarity.

It was originally thought that a group might want to be associated with a real-world location. It was for this reason that we thought it necessary to have a searching mechanism that allowed a user to find groups within a particular geo-location. Figure 3.25 describes a search that can use either a
Figure 3.24: User’s Profile Page

Figure 3.25: Location-Based Group Search
keyword or a location to find groups. This form of search was distinct from
the more traditional design shown in Figure 3.26. The latter is much more
text-heavy than the former. Later designs attempt to make the search more
appealing by utilizing visualizations.

As is the case with wireframes, the designs were not meant to be visually
appealing, but rather denote some of the key features that might take place
in the website. The following sections build upon this idea with more visually
appealing mockups.

3.3.2 Mockups

Round 1: Stylized Wireframes

The first iteration of mockups consisted essentially of stylized versions of the
earlier wireframes. These helped visualize how interactions might take place
and provided a stronger basis for arguing for or against certain features.

Figure 3.27, for example, is text-heavy, but concise. The problem with
this and other designs was that they do not take into account what to show
if a user has recently joined the website or if a user has not yet registered.
Without content to display, the functionality of the website becomes unclear,
Figure 3.27: Round 1: Home Page

Figure 3.28: Round 1: Profile Page
at least with this design. This is a problem that could still use work to develop.

Figure 3.28 is fairly consistent with the original wireframe. It is concise in that there is not much room for descriptive text, rather it contains the bare minimum content consisting of a user’s general information. Roles here are entirely user generated and are associated with modifier text that describes how experienced one is in that role.

Aside from features, it was at this point that we realized the importance of color scheme as an affect on usage. The dulled gray, red, and black color scheme left the website with a lack of vibrancy.

Round 2: Magazine-Like Interface

Mockups in the second round were inspired from magazine-like interfaces, particularly apps like Flipboard [54]. This iteration took into account the lack of imagery from the previous mockup to present a more visually attention-grabbing interface.

Figure 3.29 is much more vibrant interface but it is much more difficult to discern information from the page. Text appearing over the images makes for lack of clarity. At this point, the paradigm had shifted from being more
discussion-based to being project-based. This design was an attempt to be able to display projects in an attention-grabbing fashion, so as to encourage users to learn more about the content. The grid-like search results in Figure 3.30 seem to strike the right balance between imagery, text, and organization. Clearly defined borders between search results along with distinct sections designating a project’s name and its associated images were also noteworthy. Overlaying visualizations over text, though, suffers the same lack of clarity issues shown in Figure 3.29.

The problem with such a design in practice is that it relies on user-generated content, in this case images, to look visually appealing. Erickson and Kellogg [53] argue that social translucency forces users to behave properly within its social context—in this case behaving properly would be uploading visually appealing and descriptive images—but the cost of creating such signals might be too large. Our goal was to strike the right balance between user-generated content and website-generated content.

Round 3 or MyTri 0.5

This round of interface design also saw the first visible implementation. The reason this round was chosen as the first for implementation was because
Figure 3.31: Round 3: Home Page
Figure 3.32: Round 3: Profile Page

of its consistent color scheme, concise design, and limited need for user-generated content. Figure 3.31 displays the example landing (home) page for this version of MyTri.

While these rounds helped in displaying the end result of many of the features, they did not necessarily dictate how it was that actions were performed. For example, the profile page in Figure 3.32 displays a set of visualizations in the top half of the project page and a list of comments in the bottom half. What it does not show or feature is how to go about creating those visualization or making the comments. It is important to note that this problem is not a trivial one. Designing user interactions was going to require more iteration and testing. Moreover, this is not just a visual design issue. The lack of interaction mechanisms within the visual designs meant that they were not designed into the process itself.

Apart from the lack of user interactions, the second major point of concern was the lack of variety in the pages. In the Figure 3.32 mock profile page and the Figure 3.33 implementation of the project page, it is difficult to distinguish between the two (apart from the lack of visualizations in the latter). In order to emphasize both the project and the user, we would need to develop a design that would distinguish both. A separation of features and visual design would have aided in solving this issue.
Round 4 or MyTri 1.0

The focus of this round of mockups was to primarily work on the design. This is also the round where user testing took place, the results of which will be discussed in Section 3.4. The following images will be a mix of both mockups and screen captures from the implementation as of the time of user testing. Minor visual differences also exist between the implementation and mockup, either due to missing assets or a reconsideration of a feature. The events bar, for example, is missing from the implementation as well as the inbox page. The “general comments” sections in both the project and user pages was disabled in the implementation. In this implementation, discussion between users is only available within a project page.

Mockups will be used to display full pages, as it is more difficult to fit these pages into a single screen. Screen captures will be used to describe specific features or portions of the website that are not designed in the mockups.

Figure 3.34 is a full-page mockup of the home page, before a user has signed into the website. Here, prominently displayed, is a mission statement for MyTri. It was important for this to be the first portion of the website a user sees, since the previous designs failed to adequately describe what a user could do with MyTri. The search bar was removed from the navigation bar at the top of each page and was instead placed in the first section of the home page. This was so a user could search for other projects and people as soon as the mission statement was read. It also allowed for one less item to exist in the navbar—an area that user testing did not seem to address and that future work might include.
Figure 3.34: Round 4: Home Page
The featured projects and students sections are displayed prominently underneath the mission statement, enabling visitors to get a short summary of what is available on the website. Previous iterations required login for usage, but we felt that doing so might deter people from using the website. The inspiration for this design came from websites like Kickstarter [55], which greet a visitor with prominently featured user projects.

Figure 3.35 displays the majority of the sections of a project page that a visitor will see. Visualizations are displayed prominently at the top of the page, describing some of the project. As mentioned in Section 3.2.5, the visualizations were meant to succinctly summarize the project. Clicking a visualization’s corresponding button (e.g. “25 skills” under the pie chart) will open a section containing the information associated with it, as shown in Figure 3.36. With this design, description text is prominently displayed alongside a grid preview of a project’s images. These features did not exist in previous designs. In both the project page and the user profile page, users can edit text simply by clicking on it and filling in the corresponding form (shown in Figure 3.37). Unlike the previous designs, the prominence of the text allowed for simple interaction design.

This user profile page is a major changing point from previous iterations. Where, previously, the user page was a near carbon copy of the project page, the new user page is distinct. Editing text is similar, but the recent projects and collaborators section helps to describe relevant work of the user. The design at the bottom half (which was not implemented at the time of testing), displays small summaries of featured projects which the user has chosen to display. These were designed to aid in providing more detailed data on users’ projects without needing to leave the page.

On every page, the user has access to the navigation bar which will contain real-time notification information as shown in Figure 3.38. These notifications remain until a user hovers over them, at which point they are marked as read and no longer appear. These sorts of notifications tend to be ubiquitous across SNS platforms, most probably due to their succinctness as they only take up space if a user chooses to see them.

The search results page is one of the few that has not changed too much in the past few iterations. Projects and users are displayed within the same cards that appear on the home page, just in a grid view. More testing is needed to determine the efficacy of such a paradigm.
Figure 3.35: Round 4: Project Page
Figure 3.36: 1.0 Implementation: Skills Feature

Figure 3.37: 1.0 Implementation: Editing Data

Figure 3.38: 1.0 Implementation: Notifications
Add bio can go here. This can be stand as the "elevator speech" or perhaps a brief resume.

Description text goes here. Since MyTri’s purpose is to allow for individuals to showcase, search for, and join projects, there are currently features which allow users to create simple project pages. Members can be added to these pages after which they can write posts which other members or the public and users can see.

Description text goes here. Since MyTri’s purpose is to allow for individuals to showcase, search for, and join projects, there are currently features which allow users to create simple project pages. Members can be added to these pages after which they can write posts which other members or the public and users can see.

Projects

Molecular Biology Project
Molecular Biology Project

Contributor Discussion
Description text goes here. Since MyTri’s purpose is to allow purpose is to

Recent Projects

Collaborators

Figure 3.39: Round 4: User Profile Page
3.4 Testing and Results

Testing for the MyTri webpage took place after the completion of many of MyTri 1.0’s design features. As such, iteration from the feedback has not yet been taken into account.

The test occurred in the form of a survey available in Appendix B. Users were first asked questions regarding their age, major, and exposure to social networking sites and other web tools. For example, Figure 3.40 and Figure 3.41 both describe the users’ background in using tools similar to MyTri. Doing so helped better understand the type of user that might potentially use MyTri. If, for example, engineers were more inclined to favor MyTri’s design, it might mean MyTri would need to change its design to be favorable to more students. They were then asked to view a web page of a fake project, whose design was based a website used for a popular project at the university [56]. This site design was chosen due to its simplicity and succinctness. A more thorough survey of what sorts of websites are typical for university and graduate-level projects is something to look into for future work.

The website served as a baseline to compare with a MyTri project page. In both looking at the website and the MyTri page, users were asked to rate their interest level in a project using a five-point Likert scale (5 being the most interested and 1 being the least) as well as how effective the page in question was at describing the project.

They were then asked to create a MyTri project page of their own since doing so would allow us to gain insight into the intuitive quality of some of the MyTri features. The survey ends by asking users for further comments or suggestions.

There were in total eight participants, seven of which opted to take the survey online and one who participated in person. Ages ranged from 20 to 26, with five male and three female participants. Degree ranges included two Ph.D. students, one Master’s, two students pursuing their B.A. degree, one student in psychology and one in engineering. All were familiar with Facebook, eight out of the nine of which mark Facebook usage as “daily.”

While the number of participants was too small to be for any conclusive analysis, the responses do show certain trends in what was effective in MyTri and what was confusing or could be improved upon. For example, many participants mentioned the need for a timeline or project goals when asked
Figure 3.40: Tools Used to Share Projects

Figure 3.41: Tools Used to Find Project Teammates

Figure 3.42: Number of Users vs. Effectiveness of MyTri Project Page in Communicating Message
“what messages would you want to communicate to others when making a web page for a project?” They were then asked how effective the project page for the example project was in conveying that message. Figure 3.42 describes those results. Given that participants had already viewed the MyTri page before answering the question, answers may be biased toward what has already been seen. Future work should look at better understanding these “pain points” and addressing them in the MyTri feature set.

In terms of being an effective project sharing tool, users were asked how affective the website and MyTri page were at describing the project in question. The average change was +.375, showing a slight favor toward the MyTri project page over the web page, though there were two users who responded more negatively to the page.

The consensus on the overall layout and visual design of the MyTri website as a whole and for creating a project was that it was “simple” and “easy to navigate.” Some participants mentioned MyTri’s simplicity as a potential drawback. In reference to project pages, a student mentioned that the lack of MyTri components as compared to LinkedIn and Facebook can help “everyone find what they need easier. But this may be a downside because people might feel like it’s not enough for how they want to showcase their project.”
CHAPTER 4
DISCUSSION

4.1 Challenges

There were a number of challenges that were faced with the development of MyTri. The first, and probably most difficult, was understanding what problem MyTri was attempting to solve. Given that MyTri was at first meant to be mentor network for females, there were many discussions that led to the realization that a mentor network in the traditional sense was not going to be the focus of this project.

Relevant to this first point was the lack of user testing. While the development philosophy was aimed at being iterative, iteration using only literature research and untested designs had led to much time spent on developing a “0.5” version of MyTri that was lacking in features and usability. Now that MyTri has a groundwork of code, design, and user feedback, the focus of future development might be more effective if inclined to user testing before feature development.

From a software development perspective, the challenge is, as expected, to make working code. A suggestion on future development would be to have at least two programmers, which would make code reviews more viable. This ensures that code is kept “clean,” as another person will be forced to look at it.

4.2 Future Work

Further user testing should be the primary goal of work in the near future. The latest survey may have revealed some concerns with MyTri’s design and usability, but to be conclusive, a larger set of participants will be necessary.
Given that nearly all the participants use Facebook daily, and many use it as their primary means of finding teammates and sharing projects, it would be worthwhile to look into integrating MyTri with Facebook or other social media.

A more technical aspect would be looking into recommender systems for suggesting project pages to users. The current mechanism for related or featured projects is just a starting point. Though this feature was not explored in the testing, recommendation systems can have a large impact on usability.
As it stands, MyTri is a working prototyping for project sharing and collaboration. This thesis does not provide any assertions for the most effective method for the development and design of project-based social networking websites, but it does provide insight into the process of doing so. What can be learned from such a process is that emphasis on iterative design using research literature as feedback can only accomplish so much. It is clear that productive iteration requires input from the product’s target audience.

Even with the limited sample size that was available for testing, some conclusions can be made about the current status of the website as a usable social network. Given that the most popular method (within our small testing sample) of finding project teammates was simply in-person contact, it is clear that MyTri would be useful as a tool. Testing has also revealed the desire for fine-grained control of project page features, but also an appreciation for simplicity. Finding the right balance between control and simplicity would probably be a fruitful approach for further research.

Overall, the development process has been challenging, engaging, and rewarding. There are many challenges that need to be addressed, but with further iteration and development, refinement of MyTri into a proper tool for sharing and collaborating is possible.
APPENDIX A

CODE OVERVIEW

A.1 Server

A.1.1 app.js
The entry point for the server; start with “node app.js”.

A.1.2 config/
Contains files that either read from environment variables or contain hard-coded values for development that the rest of the application uses. Files other than index.js template.js should not be tracked by Git.

A.1.3 config/scripts/schema.sql
Contains the current MySQL schema that is compatible with the production database. Should be modified as soon as schema changes are made.

A.1.4 helpers/ and helpers/index.js
Contains a singleton object whose members are objects which interact directly with the database.

database.js
Contains wrapper methods around the node-mysql module. Use this module to start and end mysql transactions or single connections.
access.js
Singleton that contains methods which checks a user’s access level of an item in the database. If no access level is found, returns an access level of 1 (or public).

message.js
Singleton which handles queries to the “messages” table.

photo.js
Singleton which handles queries to the “photos” or “hasphotos” table.

post.js
Singleton which handles queries to the “posts” table.

project.js
Singleton which handles queries to the “projects” table.

search.js
Singleton which handles search queries.

session.js
Singleton which creates a Redis session object to be used with the Express framework.

tag.js
Singleton which handles queries to the “tags” and “hastag” table.
user.js
Singleton which handles queries to the “users” table.

queryStrings.js
Class containing query string templates used with the database helper singletons.

utils.js
Utility class that generally handles validating user input in an asynchronous format.

A.1.5 routes/index.js, routes/**/index.js
Wraps the “app.get” and “app.post” around a “route” wrapper for readability.

routes/api/*.js
Entry points for API calls. Link to the singleton database objects.

routes/web/*.js
Entry points for http requests to the web pages. Handles rendering the .jade templates.

A.1.6 views/*.jade
Template .jade files used to render HTML to the client.
A.2 Front End

A.2.1 ui/assets/less
Contains .LESS files which get compiled to CSS by a build script.

A.2.2 ui/assets/dependencies
Install location for front-end libraries. Not tracked by Git.

A.2.3 ui/assets/templates
Handlebars templates generally used in conjunction with the front-end views.

A.2.4 ui/assets/js/

config-req.js
File which configures require.js to asynchronous front-end Javascript module loading.

namespace.js
“Main” module containing that loads all dependencies.

userData.js
Module that interprets user data and returns an object containing the logged in user’s information. Currently pulls this data from a hidden element in the DOM.

A.2.5 ui/assets/js/collections
Contains Backbone.js Collection objects. Used to abstract away most API calls.
A.2.6 ui/assets/js/models

Contains Backbone.js Model objects. These are used by individual and aid in describing Backbone.js Collections.

A.2.7 ui/assets/js/views

Contains Backbone.js View objects. Renders data with Handlebars templates and will create View event listeners if necessary.

ui/assets/js/views/visualizations

Special cases of the Backbone.js views which render the visualizations in the project page using D3.js.

A.2.8 ui/assets/js/libs

Contains minified libraries used by the require.js module.

A.2.9 ui/assets/js/renderers

Helper modules that take care of boilerplate code associated with rendering Views.

ui/assets/js/renderers/editables.js

Helper module that handles identifying what DOM elements should be displayed depending on a user’s access level.

A.2.10 ui/assets/js/site

Contains entry points modules for each specific web page.
A.2.11  ui/assets/js/templates/template.js

Contains compiled Handlebars templates. Compilation is automated by a script, so this is not tracked by Git.

A.2.12  ui/assets/public

Directory containing compiled front-end assets which the Express.js app serves. Automated by a build a script, so it is not tracked by Git.

A.3  Build/Task Scripts

A task script command generally takes the form “grunt [taskname:tasksubname]”. “grunt build:develop”, for example, will set up the ui/public directory with uncompiled front-end files, for development and debugging.

A.3.1  Gruntfile.js

Setup script for the grunt task manager that tells it to look for configuration scripts in the build/tasks directory. Should not need to be modified.

A.3.2  build/tasks

default.js

The default command to be run if the “grunt” command is executed without parameters. This should be set to whatever is the most convenient for the developer.

build.js

Commands associated with the compiling or building files. The two most common will probably be “grunt build:watch” for building and compiling front-end files as they are changed during development and “grunt build:production” for making files production ready on one of the Git production branches.
tiny.js

Custom grunt task to run grunt tasks asynchronously. Primarily used for building production files and pushing them up to the production server.

A.3.3 build/tasks/options

Grunt options for grunt task modules. The Gruntfile.js will look at this directory and name grunt tasks by their file name. For example, clean.js will be referred to as “grunt clean”.

A.4 Package Management

A.4.1 package.json

JSON file containing a list of dependencies for the node package manager. To install all dependencies run “npm install.” This file should be kept up to date with the latest non-breaking libraries. Heroku will look at this file to cache dependencies on the production server.

Packages are installed in the node_modules/ directory and should not be tracked by Git or other version control.

A.4.2 bower.json and .bowerrc

Similar to package.json, but keeps track of front-end packages and installs them in the directory specified by .bowerrc.

Currently points to ui/assets/dependencies. This folder should not be tracked by Git or other version control. To install dependencies run “bower install”.

57
A.5 Miscellaneous

A.5.1 Procfile

File used by the Heroku app to determine launching point for the application.

A.5.2 .env

Contains environment variable information. Should NEVER be tracked by Git or other version control as it contains sensitive database information and API keys.
Survey and Questionnaire for MyTri Usage

Background
What is your age?

What is your gender?

What degree (if any) are you pursuing?

I use the following social networks (check one for each row):

<table>
<thead>
<tr>
<th></th>
<th>N/A</th>
<th>Rarely</th>
<th>Monthly</th>
<th>Weekly</th>
<th>Daily</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LinkedIn</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meetups</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Behance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What do you use to find project or group teammates (circle all that apply)?

Facebook    LinkedIn    Meetups    Contact in class/in person
Other (please specify):

What have you used to share projects (circle all that apply)?

Facebook Group/Page    LinkedIn    Website    Github    Behance
Wordpress/or similar    Other (please specify):
Viewing the Website
In your own words, briefly summarize the project (1 to 2 sentences).

How interested are you in learning more about the project (circle one)?

<table>
<thead>
<tr>
<th>Very Uninterested</th>
<th>Somewhat Uninterested</th>
<th>Neutral</th>
<th>Somewhat Interested</th>
<th>Very Interested</th>
</tr>
</thead>
</table>

How effective was the website in describing the project (circle one)?

<table>
<thead>
<tr>
<th>Very Ineffective</th>
<th>Ineffective</th>
<th>Neutral</th>
<th>Somewhat Effective</th>
<th>Very Effective</th>
</tr>
</thead>
</table>
Viewing the Project Page Infographic
In your own words, briefly summarize the project (1 to 2 sentences).

How interested are you in learning more about the project (circle one)?

<table>
<thead>
<tr>
<th>Very Uninterested</th>
<th>Somewhat Uninterested</th>
<th>Neutral</th>
<th>Somewhat Interested</th>
<th>Very Interested</th>
</tr>
</thead>
</table>

How effective was the website in describing the project (circle one)?

<table>
<thead>
<tr>
<th>Very Ineffective</th>
<th>Ineffective</th>
<th>Neutral</th>
<th>Somewhat Effective</th>
<th>Very Effective</th>
</tr>
</thead>
</table>

The visualizations/web page were appealing to look at (circle one).

<table>
<thead>
<tr>
<th>Strongly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Agree</th>
<th>Strongly Agree</th>
</tr>
</thead>
</table>


What messages would you want to communicate to other viewers when making a web page for a project?

The visualization/web page was effective in conveying that message (circle one).

<table>
<thead>
<tr>
<th>Strongly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Agree</th>
<th>Strongly Agree</th>
</tr>
</thead>
</table>

What information was conveyed in the project page effectively?

What information was missing in the project page that you might want to have known?
Creating a Project Page/Infographic

What information (if any) was not conveyed in the project you created?

Describe your experience with creating a project.

Describe your experience with MyTri as a whole.

Any further comments or suggestions?
REFERENCES


