PauseLab: Participatory Budgeting in Charlottesville

Henry Garrett
University of Virginia
hmg8je@virginia.edu

Amir Gurung
University of Virginia
ag5ct@virginia.edu

Sugat Poudel
University of Virginia
sp5pe@virginia.edu

Sudipta Quabili
University of Virginia
ssq9tf@virginia.edu

Oscar Sandoval
University of Virginia
oks2vd@virginia.edu

Jay Sebastian
University of Virginia
jgs3cd@virginia.edu

ABSTRACT

We create a system for PauseLab to manage community engagement in proposing and selecting public art projects. The system serves three primary purposes: collection and exposition of ideas, compilation of proposals to and from the community, and facilitation of community input on which ideas will receive funding. Through our system, residents of Charlottesville will now be able to streamline their voices in participatory budgeting that directly affects their community. Each step of the participatory budgeting process is automated in the collection and archiving of data, and each unique user role plays a part in ensuring its progression. At the end of the participatory budgeting process, blog posts inform the community of the progress of the proposal select by its residents. PauseLab resets the process every year with a new round of funding, and our system ensures it will be a seamless transition to the next budgeting process for benefiting the Charlottesville community.

1 INTRODUCTION

The objective of our Computer Science Service Learning Practicum project is to design a system for Charlottesville-based non-profit, PauseLab, that would facilitate projects in Charlottesville-area communities. PauseLab is an organization that promotes civic-engagement in the Charlottesville area by sourcing ideas for projects from the residents themselves. These projects range from setting up traffic lights to building public art pieces.

PauseLab is a Charlottesville-based non-profit that looks to facilitate civic engagement in the community. For their BeCville initiative, they were awarded a $15,000 grant to use on community projects. Rather than just initiate the projects on their own, they have decided to look toward the community for ideas. Therefore, they have employed a system called participatory budgeting where the residents themselves pitch ideas for projects and then also vote on these ideas to select a number to fund and actually carry out.

In the past, PauseLab has used multiple platforms in order to handle the participatory budgeting process, from using WordPress sites and paper forms to gather ideas and votes, to using Excel files to handle data organization and storage. The participatory budgeting system will handle everything starting from the idea collection phase, where residents of the Charlottesville area will be able to submit location-based project ideas. The system will then move over to the proposal collection phase, where artists will submit proposals in response to ideas that interest them, so that they may helm the projects if they ever come to fruition. Voting will then be done through the site as residents will be able to select proposals that they support, and the proposals with the highest number of votes will be funded and completed by the community. The site will also keep residents updated throughout the process so to keep everyone well informed and involved.

Ideally, this system will streamline PauseLab's idea collection process. They will no longer need to rely on disconnected platforms and physical media as everything will be consolidated into one digital platform that will keep track of everything intuitively and effectively. This also means there are also now tools in place to gather analytics and more information about the community. Overall, we hope that the system will foster community involvement in Charlottesville and make the project funding process more democratic.

2 BACKGROUND

Participatory budgeting is a 20th century democratic innovation that was started in Porto Alegre, Brazil. It is a process by which community members, rather than elected officials, decide how to allocate and spend part of the local budget. Since its inception in Brazil, it has spread to over 3,000 cities, including Charlottesville, VA. This novel form of inclusion hopes to broaden the scope of representation in political decision making processes by increasing democratic access to historically marginalized groups.

Participatory budgeting can be divided into three distinct phases: deliberations, decisions, and implementation. Deliberations is composed of local assemblies, where any citizen can come together to participate and vote in order to determine local fiscal priorities. This can be modeled as collecting ideas from local residents through online or offline mediums, and is highly dependent on the effectiveness of engagement strategies

that the governments utilize. Decisions relates to the final decisions regarding the allocations of the public budget by the city officials. This is mostly an internal process, however it is driven purely by citizen input. Lastly, implementation boils down to monitoring the implementation and state of the selected public projects at the hands of the city.

Participatory Budgeting (PB) has spread to many local governments in the U.S. including municipalities in New York City, Long Beach, and Charlottesville, among others. The PB implementations in these municipalities rely on existing and deep rooted local institutions such as schools and community groups to perform outreach regarding the PB events. Moreover, they are slowly starting to use digital tools to supplement engagement, collecting citizen input and internal management of data.

3 RELATED WORK

PB Long Beach is a system for collecting ideas from the community based on its geographical idea. PB Long Beach's main interface is the interactive map is where users can place where an idea should be enacted. After the marker is placed, users can further elaborate on what the idea entails, primarily the premise, how it helps the community, and what category it best fits under. Users can optionally fill out demographic information to help PB Long Beach better understand what user groups need what. After ideas have been approved, they show up on the front page map, and users can further discuss them them in their respective comments sections.

PB Stanford is a system for collecting votes on projects that the public budget can be spent on. The main feature of the system is the selection of some number of projects whose combined cost is under the max budget available. The left toolbar shows all the proposed projects that can be selected, and the main page allows the selection of ideas with a brief description of what the project is, how much it will cost, and where the project will be enacted. If some number of projects have already been selected or the total cost of the budget has been exceeded, no more projects can be selected. After users submit their votes, their some identifying information is then filled out to ensure that they are eligible to vote in the Long Beach area.

PB Long Beach and and PB Stanford are both systems that excel at what they do, but are too specific for PauseLab's use. PB Long Beach can only be used to collect ideas from the community, and PB Stanford can only be used to aggregate votes from the community. Both systems lack the aspect that makes PauseLab unique: the emergence of artists that submit proposals that will benefit the community. In traditional participatory budgeting, the approved projects are fulfilled by third-party contractors, and the community holds the contractors accountable for the completion of the projects. PauseLab instead makes a call for artists to submit proposals for the community based on the ideas the community is trending towards. After the proposals have been approved and voted on by the community, the artist who proposed the idea will then be assigned for the project's completion. Milestones in the project's completions fall under the blog posts that the artist releases monthly, which is another feature that PB Long Beach and PB Stanford lack. For PauseLab's system, our ultimate goal is to consolidate all these features into one place where an administrator can seamlessly change between their phases of participatory budgeting.

4 SYSTEM DESIGN

The PauseLab project uses a Ruby on Rails framework with a MySQL database. We used various gems (Ruby libraries) and a few client-side libraries to accomplish pre-solved tasks such as authentication, authorization, file attachments, and map displays. At a high level, our application is just a schema exposed by templates and glued together with gems and framework-related code.

4.1 User Roles

The application is designed around six different roles that a user can take. The roles are mutually exclusive, in that a user can only occupy one at a time.

4.1.1 Admins. Site administrators who have the highest privileges, including creation (and deletion) of other users in other roles. They control and maintain the system.

4.1.2 Moderators. Trusted users who are permitted to manage any data in the system unrelated to users (e.g. ideas, proposals, blog posts). They revise and filter site content.

4.1.3 Steering Committee. Users who are permitted to see most
data in the system, even if it not accessible to residents (see below), and are able to use this data to communicate with each other. They direct PauseLab's focus and draw insights from the community. *4.1.4 Artists.* Users who propose public art projects to address community needs.

4.1.5 Super Artists. A subset of artists who are selected to implement projects and report on their progress. They carry out the will of the community.

4.1.6 Residents. Users to view the system's published content and may submit ideas for improvements to their area. They are community representatives.

4.2 Application Phases

There are several steps in the chronology of one cycle (or as it is known in the code, an “iteration”) of the project. The first phase is ideas collection, where residents may submit ways that they perceive the community can improve. The next phase is proposals collection, where artists can submit more fleshed out plans that address the previously submitted ideas. The last phase is voting, where residents submit votes on which artist proposals should get funded. Once voting is complete, some number of

projects are selected to be implemented, and users can visit the site to view news on progress.

4.3 Models

There are several models in the system that are implemented through a relational database schema (i.e. SQL); some are related to the project-specific tasks, while others are linked to auxiliary tasks like authentication and authorization.

4.3.1 Idea. Submitted by residents in the first phase of the application. Contains freeform text about the idea itself, a category that the idea belongs to, a geographic location (i.e. latitude and longitude), and some personal information about the resident submitting the idea, including name, email, and phone number. When an idea is first submitted, it has an "unchecked" status, and is not publicly available. Only when an admin or moderator approves the idea does it become visible.

4.3.2 Proposal. Submitted by artists in the second phase of the application. Contains freeform text about the proposal itself, a title, a link to the artist's portfolio. Similar to ideas, proposals must be approved by admins or moderators before being publicly visible.

Proposals may also be marked as "funded" to indicate to users which ones were selected in the voting phase.

Proposal Budget. : Contains cost fields for various parts of the plan. This encourages artists to think about how they will allocate their funds if their project is selected.

Proposal Comment. : Contains text for internal comments between members of the steering committee regarding a proposal.

4.3.3 Category. Created by moderators as a means of grouping ideas and proposals together. Contains a short piece of text. Some example categories include: "Health and Safety," "Art and Culture," "Infrastructure," and "Other."

4.3.4 Vote. Submitted by residents during the last phase of the application. Has a many-to-many relationship with proposals, as well as personal information on the voter such as name, email, phone number, and address.

4.3.5 Blog. Created by super artists or moderators during any phase of the application, but particularly after voting is complete, to report any news about PauseLab. Contains a title and freeform text body that may be embedded with formatting, images, or hyperlinks.

4.3.6 Iteration. Captures the notion of cycles of the application. PauseLab can conceivably operate the ideas-proposals-voting process for community projects at least once per year, so each time it needs to reset it creates a new iteration. Each iteration has many ideas, proposals, and votes, and when each of these three models is shown on the site they are enumerated according to the current iteration.

4.3.7 Landing page. Created by moderators for specific but mutable page content. Contains freeform text that may be embedded with formatting, images, or hyperlinks. Landing pages are exposed on the About page, the Ideas collection home page, and the pages that artists and steering committee are redirected to once they log in.

4.3.8 Mass Email. Created by moderators for sending emails to certain role groups. For example, a moderator may want to send out an email to all artists reminding them to send in proposals before that phase of the application is over.

4.3.9 User. Contains the profile of users who authenticate into the system, including name, email, password, phone number, role, and an avatar image. Users may be associated with proposals, proposal comments, and blogs. However, they are not associated with ideas or votes because those submissions would typically be made by "one-time" users; the personal information collected in those submissions reside in those models themselves. *4.3.10 Ability.* Defines the permissions for each user role. Roles generally fall into a hierarchy where those further down the chain possess a subset of permissions belonging to those higher up. Admins are at the top, followed by moderators, steering committee, super artists, artists, and residents, in that order.

- Admins: Can manage (i.e. create, read, update, destroy (CRUD)) any type of data
- Moderators: Like admin, but cannot manage user data
- Steering Committee: Can create proposals, proposal comments, ideas, votes, and manage anything that they created. Can read all proposals, regardless of approval status. Can only read approved ideas. Super artists: Like steering committee, but can create blogs, and cannot create proposal comments. Can only read approved ideas and proposals.
- Artists: Like super artists, but cannot create blogs.
- Residents: Like artists, but cannot create proposals.

4.4 Libraries

For most models (i.e. Idea, Proposal, Vote, Blog, Landingpage), much of the Rails code was boilerplate. We simply generated routes and controller code for the CRUD operations and defined the permissions (i.e. which users could take which actions) in Ability. The templates were designed to be clean and functional, and parts of the HTML were generated based on authorization. A few of the more interesting models and the gems used to address them are described below.

For user management by an admin, we created routes and controller code similar to typical models. However, the User model connected to the Devise gem to allow for some easy-to-use methods regarding authentication (e.g. checking if a user was signed in and automatically hashing passwords). Devise also allowed us to create forms for users who sign up for

accounts themselves and automatically handled common user management tasks like password reset.

For user authorization, we used CanCanCan to enforce the permissions declared in Ability. For the CRUD operations this gem built authorization in automatically, such that if users do not have permission to see or do something, they are redirected to the root URL with an error message. The view templates used simple helper methods inside if-statements to only generate links or buttons that a user was authorized to follow or press.

Some models (i.e. Blog, Landingpage) necessitated the use of a "what you see is what you get" (WYSIWYG) editor, where the user could format text (size, style), insert images, and create hyperlinks without typing out HTML. To accomplish this we used Trumbowyg, which transformed <textarea> elements into easy-to-use GUIs.

To avoid automated bots from spamming certain form submissions (namely, user account creation and votes), we added Recaptcha (a CAPTCHA service provided by Google) as an extra validation step. The user needs to check a box and in some cases perform another task to prove they are human. A Recaptcha element can be embedded into the HTML and verified in controller code with some simple Ruby helper methods.

For internationalization, we used the i18n library native to the Rails framework. Any strings that are part of templates are indirectly referenced through a call to a translate method which looks up a key-value mapping in a YAML file. By creating different YAML files for each language and setting a locale variable in the request parameters, the proper string is shown on the page.

For several templates that show an instance of a model intended for public consumption (i.e. Idea, Blog), a Disqus thread is embedded at the bottom of the page. This platform allows users to post comments after they authenticate through a common service such as Google (or create a Disqus account). Even though this separates the application-specific account and the common one (possibly inconveniencing users who may need both), it captures a demographic that may want to contribute to the discussion without investing time into making a new account.

The final significant library is Google Maps, which is embedded in pages that display ideas. Since an idea contains coordinates about the area it addresses, Google Maps is an intuitive way of allowing location input on a form and then displaying that location to the public. Each approved idea shows up as a marker on the map with an icon that corresponds to the idea category. Zooming and panning features are built into Google's Javascript code.

A variety of other libraries were used, including Paperclip for file attachments, Simple Form for easy form code, Bootstrap and Font Awesome for clean, elegant, and responsive styles, Rubyzip for zipping multiple files into one download, and Chartkick for producing line graphs and pie charts on a webpage.

5 PROCEDURE

With the overall architecture explained, we will elaborate on system design on a per-phase basis. For each phase, we will describe how a user of each role interacts with the system.

5.1 Idea Collection

The first stage is idea collection. During this time, residents are able to submit project ideas by completing a form and choosing a location. Upon submission, a flash message will appear and the resident will be taken to a screen that displays all approved ideas. Additionally, an email is sent to the resident to assure them that their idea is pending approval. Moderators will take those ideas and approve/disapprove them. Once approved, they appear on the homepage for all residents to view. Approved ideas can be shared via Facebook or twitter. Residents may also "like" ideas to indicate early interest.

5.2 Proposal Collection

Once all ideas are collected, the admin will change phase from ideas to proposals. During this time, artists are called to submit proposals. The home screen displays all approved ideas, and residents can begin to like these ideas. Artists will create proposals based on popular suggestions. A proposal is composed of a title, description, a small essay on why the artist believes it will benefit the community, expense fields for the projected cost of the proposal, and artist credentials to prove that they are capable of completing the project. Afterwards, an email is sent to alert the artist about the status of their proposal. Like ideas, proposals need approval from moderators/admins as well. Steering committee can make comments on proposals which are not shown to the public. Unchecked proposals

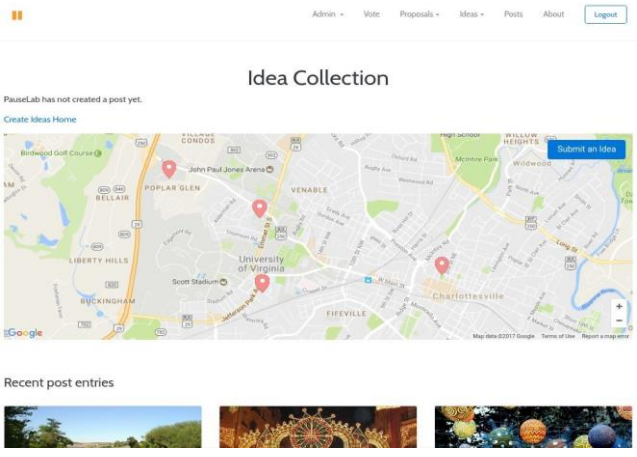

Figure 1: Home page during ideas phase

remain unseen to residents. Once the phase changes to voting, residents will see all approved proposals. Meanwhile, the home page will display all current ideas. Users can still like ideas during this time, but they may not submit ideas.

5.3 Voting

After the admin changes phase from proposals to voting, the home screen changes once more. All approved proposals will be displayed for residents where they can choose three proposals to vote for. To do this, the resident must fill out the voter ballot stating their name and contact information. They must also check the voter pledge saying they are within the residence they are voting from. Before the vote is submitted, the voter must verify the Recaptcha to assure the system that that it is not a fraudulent vote.

The proposal with the highest number of votes will become the new community project. Artists that are associated will become promoted to super artists. With this, they can create news posts that update residents about their project status. These posts can be seen from the home page as well, where it displays three of the latest posts.

5.4 Iteration Complete

After all three of the phases are completed, the admin can end the iteration. This system flushes out the information associated (ideas, proposals, votes) and creates a new iteration. Users are not affected and remain present within the site. Within the iteration console, the admin/moderator can look at past iterations and export information about that iteration as a zip file. Super artists will continue to update residents about project status as a new cycle begins.

6 RESULTS

Some of the main issues PauseLab wanted to address were creating a way to vote on projects online, increasing overall participation in

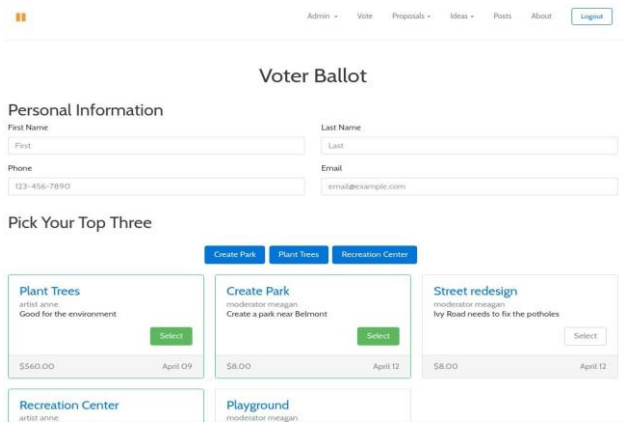

Figure 2: Home page during votes phase

idea submission and voting, streamlining project proposal submission, and allowing steering committee members to easily view and comment on proposals. In the past PauseLab only offered an online form to submit an idea. Proposal submission was only through a link to various Google forms, and all voting was done with paper ballots. Our system increases resident participation by giving a more visual representation of ideas and proposals. Artists will have tools to view what the community wants, and can easily create proposals based on those ideas.

With online and mobile voting, as opposed to paper voting, we estimate the number of votes per cycle to double. During the current cycle for PauseLab there were 250 ideas submitted on their website or in paper form. We estimate the number of ideas submitted will increase by 50%. The system will also save the steering committee's time by centralizing their communication as they deliberate on a proposal. Other data is also centralized and does not require multiple spreadsheets. Regarding improved time efficiency, a client at PauseLab said, "[the] site will drastically change the amount of time we spend on emailing and sharing information."

7 CONCLUSION

The goal of our developed software was to help meet the needs of PauseLab. They wanted a better way for residents to get more involved with the participatory budgeting system in Charlottesville. Our system gives residents a comfortable and accessible way to take part in the process, and it also gives special user roles all the tools they need to complete it. With the new BeCville system, ideas are shown in a more logical manner on a map rather than a spreadsheet. Voting used to require paper ballots being collected from all over town, but now it can be done on PauseLab's own site. In the past email chains and phone calls were required to communicate between steering committee members, but now comments can be made privately directly on the proposals.

Our system had to be easy to use for residents. Participatory budgeting is reliant on residents of a community actively participating, and ease of use plays a large role in continued participation. We made the system simple and visually pleasing in order to keep people coming back every iteration.

Participatory budgeting is a complicated process with many different users, phases and functions. The components of our system bring all of these factors together in a logical and convenient way. Using many different gems and libraries, we created trustworthy and reliable functionality to help PauseLab continue to improve Charlottesville. With this system more people will get involved in idea submission and voting, and as a result allocation of the public budget will become a more democratic process.

8 FUTURE WORK

Further implementations could improve on analytics, aesthetics, and security. Currently, there are basic administrator analytics

to see the number of ideas submitted and categorical ratios. As the site progresses, we could use those analytics to understand how users interact and better design the system. For example, if there is a certain time of year where ideas submission has low participation, then we can possibly alert moderators and administrators that the site is not highly active. Or if one category becomes highly popular, there could be another alert to administrators about how the current categories are ranking within the system.

Security improvements could be made on "likes" for ideas. The current implementation relies on cookies to prevent multiple likes. However, removing the cookie will allow for a user to create multiple likes. If we had more time, likes would become easier and more convenient for users. One possibility is that residents could see ideas, but in order to like them, they would have to login through a service such as Facebook.

Another improvement could be made on data management. Our system works well for small-scale applications, but it is unclear how it will manage on a higher scale. Changes to server infrastructure, such as load-balancers, may be necessary. In addition, the EC2 instance is launched under the free version. If PauseLab continues to grow at a rate that no longer supports this, then changes to the instance will become necessary.

ACKNOWLEDGMENTS

We would like to thank Matthew Slaats for his dedication towards the project. It would not have been complete without his guidance and vision.