**Testing and UI/UX**

This project started off with a simple idea of a forum where users could post questions and answers about bugs they found in their area. We started off with a simplistic barebones html site just to show content and layout and to get us off the ground (see Figure 1).

Users we tested shared feedback that the site navigation was clear, but the intent and color scheme was a little lacking, especially the first choice we settled on, a garish green (see Figure 2) that worked well on some screens but awfully on others. This was due to the fact that we just had “green” instead of a specific hex so that was taken into account in our next step. Users also mentioned our lack of mobile friendliness as during testing, most users pulled our site up on their phones or tablets. This made it clear that we needed a more mobile-friendly navigation bar and site overall.

Based on their feedback, we went back to the drawing board and created a wireframe (see Figure 3) using FluidUI to experiment with complimentary color schemes and settled on #6F94D1, #D1AC6F, #6FD17B. We also developed an icon for site recognition/branding that incorporated the stackoverflow icon that our site is based off of and a starkly contrasted colored bug, the ladybug (see Figure 4).

Upon researching successful e-commerce site and platform, Etsy, we found that the simplicity of their stack would work well with what we intend to accomplish. Etsy is programmed via PHP on Apache probably due to website development versatility. They use MySQL, cache layers, and F5 load balancers to maintain data flow between sellers, buyers, and Etsy (Limpalair, 2016). Our current site is coded with HTML5/CSS and PHP to work seamlessly with our eventual SQL database integration. We also implemented a skeleton CSS framework that we found to work well aesthetically as well as in a mobile-first design by first applying Eric Meyer’s reset.css and then applying the skeleton CSS on top of this.

Our current home page has a ways to go before looking like our wireframe site, but the basic layout remains intact (see Figure 5). We wanted the site to be simplistic, mobile-first, and intuitive to use to encourage user interaction/postings. Our home page will ultimately list popular “top” questions that are most commonly asked or highest rated according to customer rating as well as click-frequency. We chose the Google font Droid Sans due to its easy readability on big and small screens alike.

The sign-up page features common and interest-piquing questions to encourage membership (see Figure 6). While users can anonymously browse questions and answers on bugoverflow, in order to post or answer a question, they must have an account. Buttons for members or non-members not logged in or signed up will take users to the login page that will feature a message saying “Please sign in to post or reply”. The login page will also say “Not a member?” and have a button to take non-members to the sign-up page. The idea behind these choices is two-fold: (1) to encourage membership levels, but not to make the user think about joining as a chore but as a natural and logical extension of visiting the site (2) during user testing, it was noted that there was a scramble for some testees before clicking on the sign up or login button (dependent upon their membership status) in the navigation bar, so naturally, these links had to be placed in more logical areas to account for different user logic.

Like the sign-up page, the login page is simplistic with only the minimal required fields to log in (see Figure 7). We want users to get back to using bugoverflow as fast as possible. A simple login page serves as a brief interruption to unlock a wider use of resources, not as a space to lure or more likely, annoy, users back with flashy images or graphics.

Our current question and answer page (see Figure 8) does not yet look similar to the wireframing image mentioned before at this point in the project so figure 9 UI choices will be discussed instead. As seen in figure 9, the question and answer page still features the simple fields common to the rest of the site, but with a few more necessary features including image upload and geolocation tracking. During user testing, there was a lot of back and forth concerning the issue of privacy in terms of automatic geolocation tracking. Some users found the geolocation feature to be convenient, saving them the effort of typing in and/or recalling where the bug was found. Others argued that they should be able to type in their location in the case that they moved from the original location the bug was spotted in and did not want a site knowing location without permission. To satisfy both parties, automatic geolocation was turned off and an optional location text field was added in the case that users did not want their location to be known. In the case that no locational information is submitted for the post, location data will be extracted from the photo upload itself. When users select submit, they are agreeing to the release of this information, therefore consenting to a privacy agreement. These errors and flaws we run into from user testing are key to our project’s success. As stated in *The Devops Handbook*, organizations need “a dynamic system of learning that allows us to understand our mistakes and translate that understanding into actions that prevent those mistakes from recurring in the future” (Kim et al., 2016). Testing and making mistakes is vital to preventing future mistakes, ideally before live deployment.

To create valid test data for usage testing, we intend to implement the javascript library of [faker.js](https://github.com/marak/Faker.js/) to generate fake names and accounts. Part of the syntax consists of the following:

**var user = {  
 name: faker.name.findName(),  
 email: faker.internet.email(),  
 bio: faker.lorem.sentence(),  
 image: faker.image.avatar(),**

**faker.helpers.createcard();  
}**

The helpers library will create full contact cards including name, email, bios, images, and blog posts to reflect how the site will be engaged once more tables are populated. We currently have about 10 accounts generated from our security team to get testing off the ground for UI/UX choices, but plan to implement at least 40 more. The data we have in our database will be real and valid since we don’t want to corrupt our original data. With users’ ability to add data into our forums, the chance for corruption will already exist so to start user testing we have only factually accurate bug information. As Gene Kim says in his keynote speech (Kim, 2014), the accrual of technical debt leads to fragile artifacts. To avoid the downward spiral of our site, keeping data uncorrupted and organized is key. Eventually our site will need monitoring in the case that inappropriate or inaccurate content is posted. This is the only way to ensure that the most critical, possibly revenue-generating operations of bugoverflow remain less than fragile. Avoiding compounding technical debt by learning from and preventing future problems improves our site and consumers’ experiences and success.

Moving forward, ideally, automated testing would be helpful for site improvement be that in terms of UX, database, or general programming deployments. *The Devops Handbook* supplies a model (see figure 10) that works successfully for Google that we will consider adopting when it comes to test builds in bugoverflow. They have their commit stage to automated test stage automatically approved while exploratory tests to production deploy are manually approved (Kim et al., 2016). Though our test times won’t be nearly as fast, it is still a strong and proven successful model for strive for.

**References**

Kim, G. (2014, May 19). *DevNation 2014 - Gene Kim - Afternoon Keynote: Why Everyone Needs DevOps Now* [Video file]. Retrieved from <https://youtu.be/vTxAR8EOc8A>.

Kim, Gene & Humble, Jez & Debois, Patrick & Willis, John. ( © 2016). *The devops handbook: how to create world-class agility, reliability, & security in technology organizations.* [Books24x7 version] Available from <http://common.books24x7.com.ezproxy.gl.iit.edu/toc.aspx?bookid=117823>.

Limpalair, C. (2016, March 23). What does Etsy’s architecture look like today? Retrieved from <http://highscalability.com/blog/2016/3/23/what-does-etsys-architecture-look-like-today.html>.

**Figures**

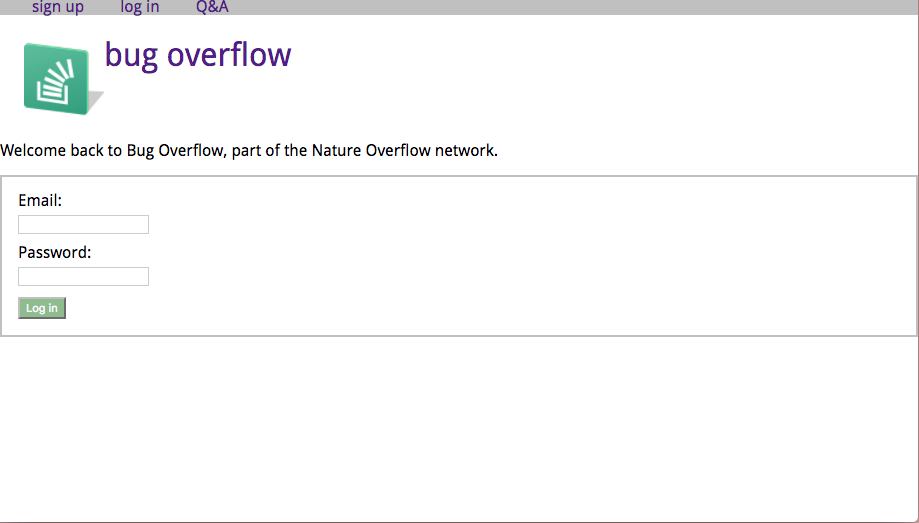
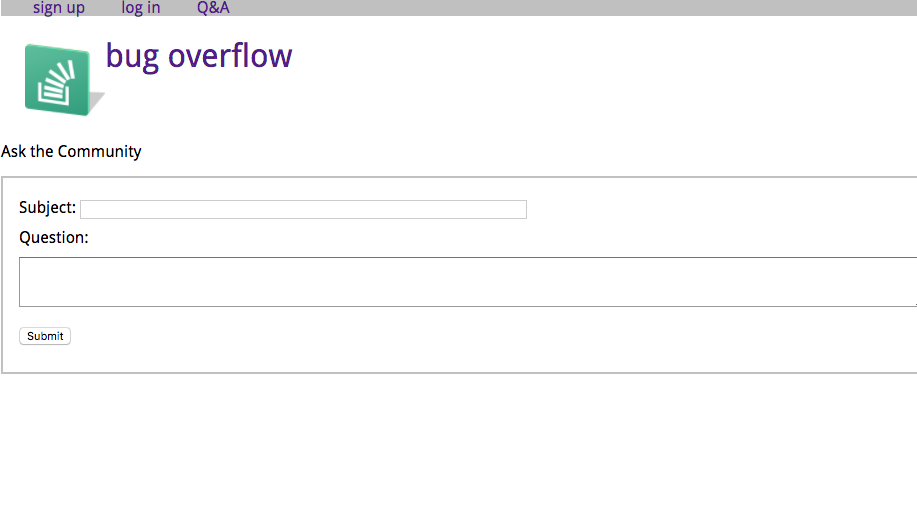
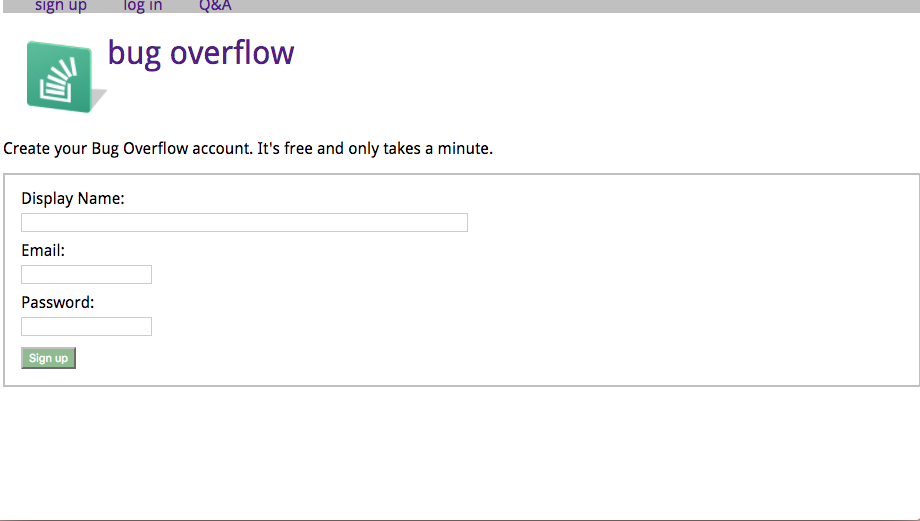
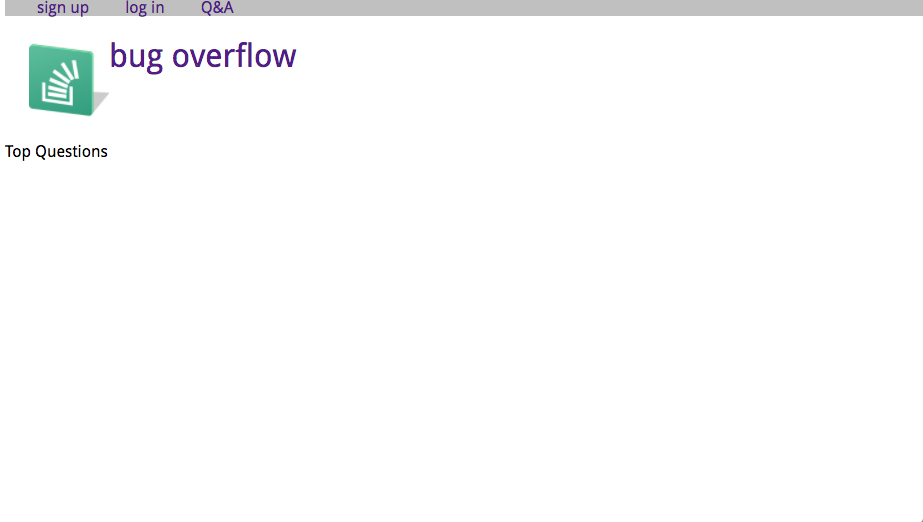


Figure 1, Basic HTML of bugoverflow site

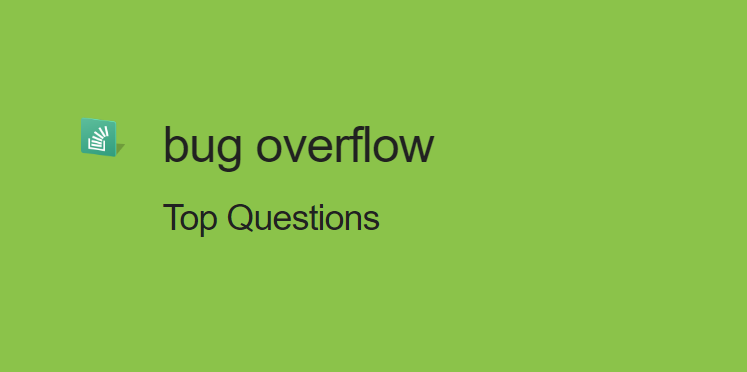


Figure 2, Implementing color/CSS to basic site

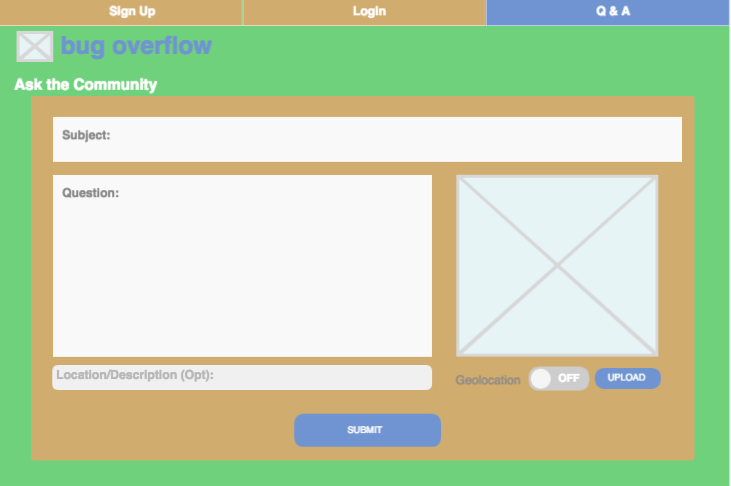
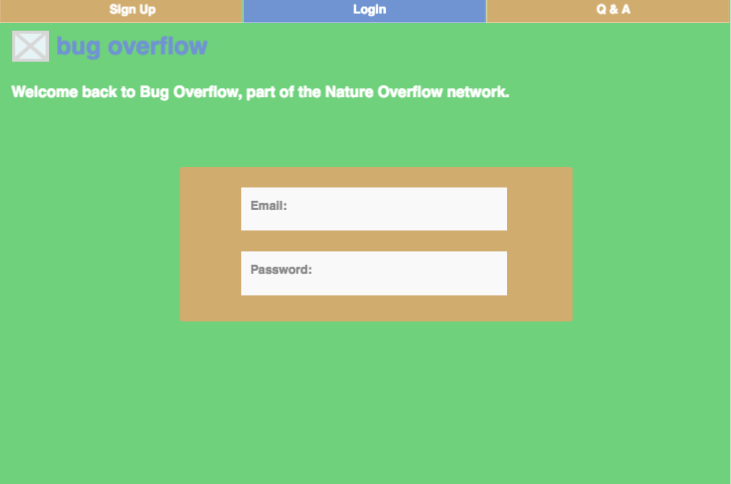
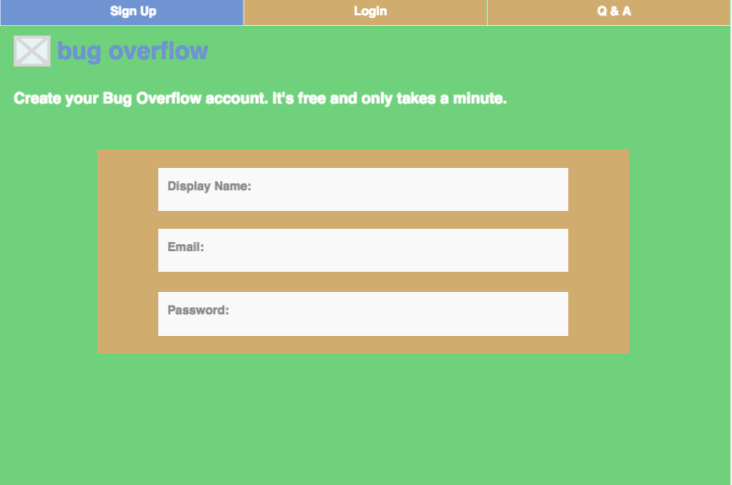


Figure 3, Wireframing and final color/design choice for bugoverflow site

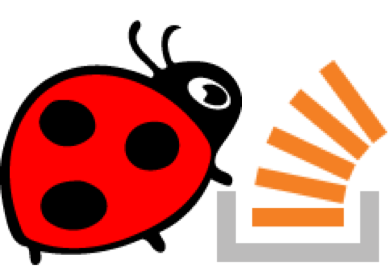


Figure 4, bugoverflow icon integrating memorable bug pushing an overflowing stack taken from stackoverflow icon

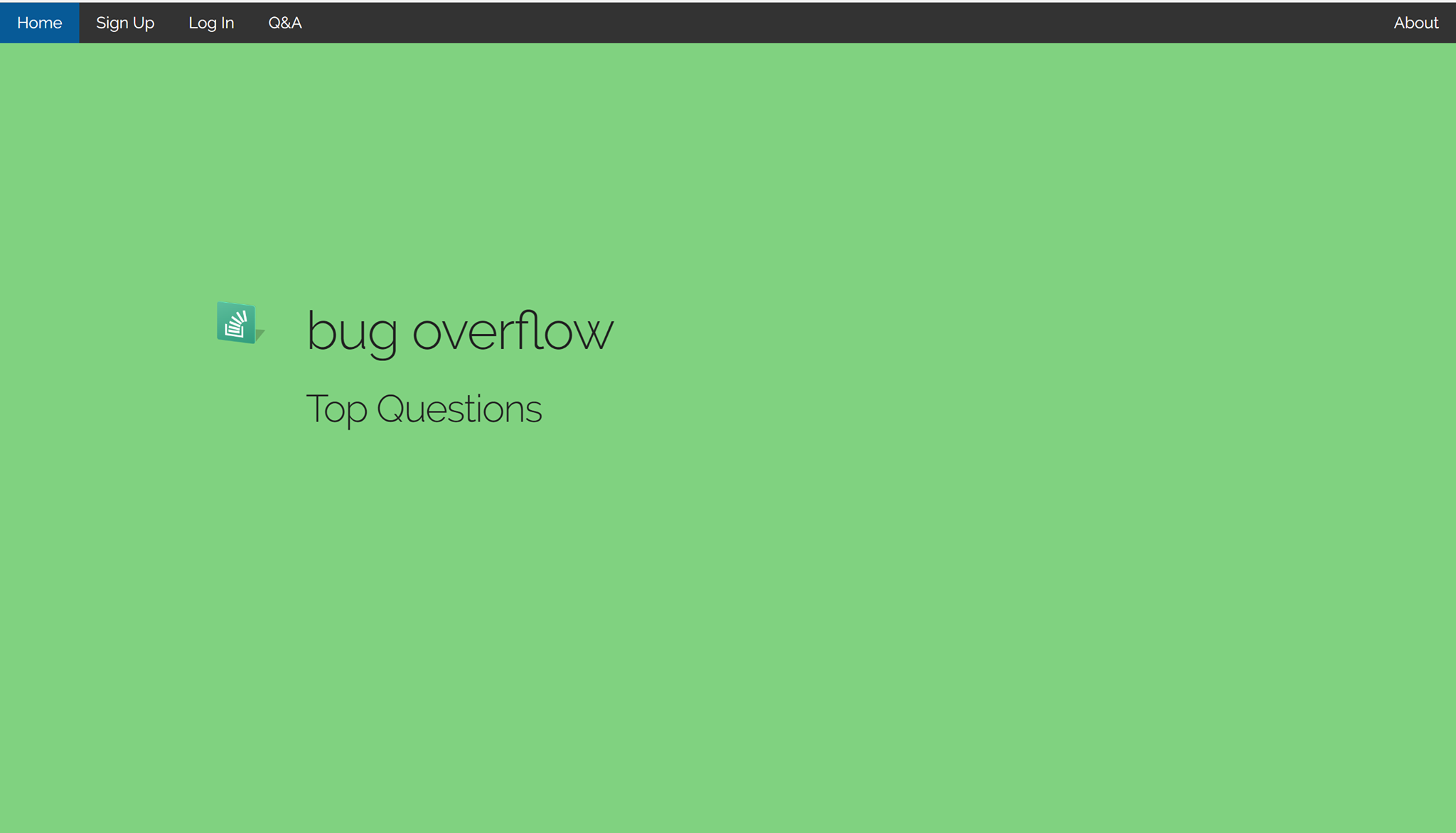


Figure 5, mobile-first design home page during mid-point in project

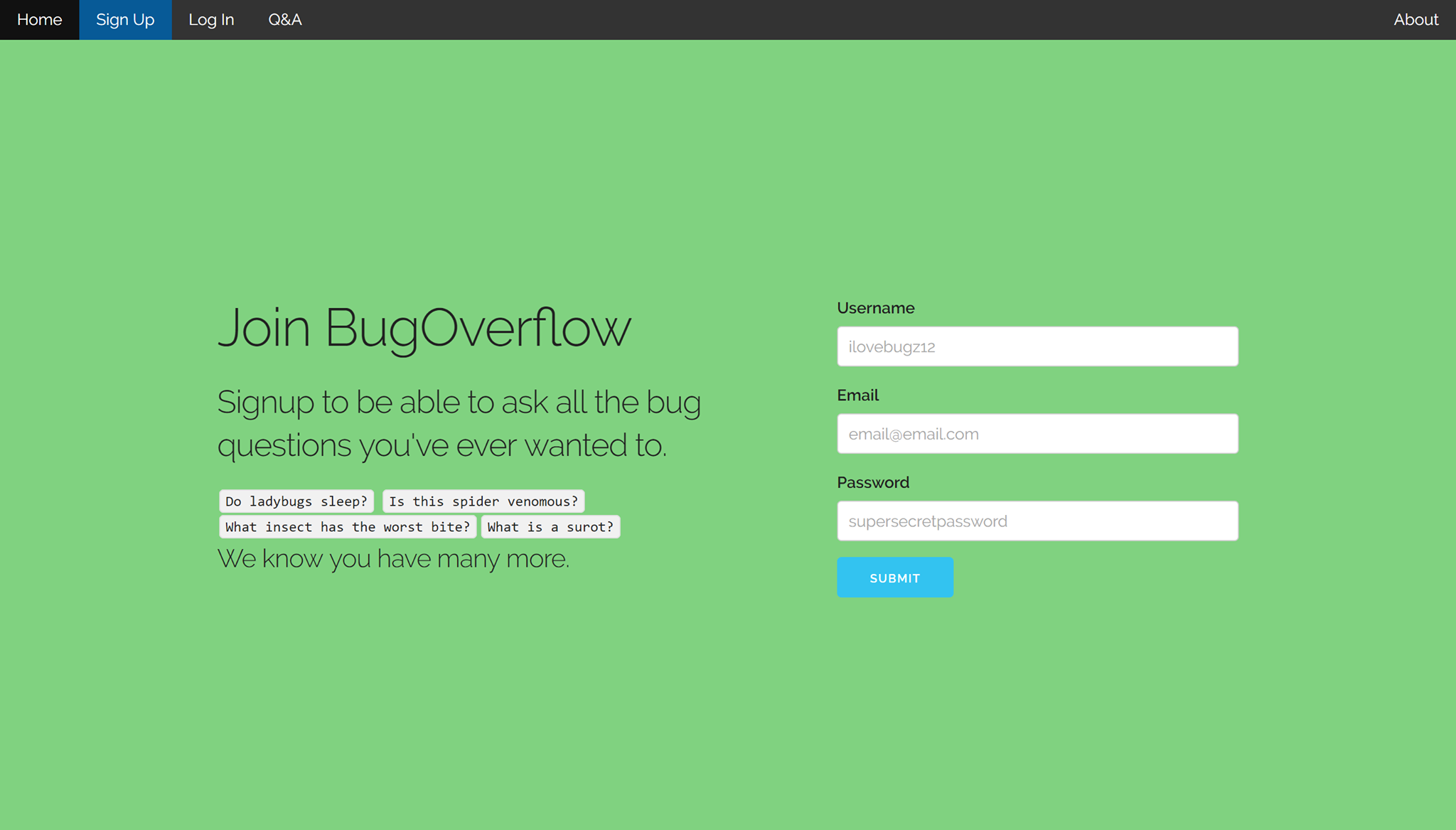


Figure 6, Sign-up page with minimum fields to encourage easy membership access

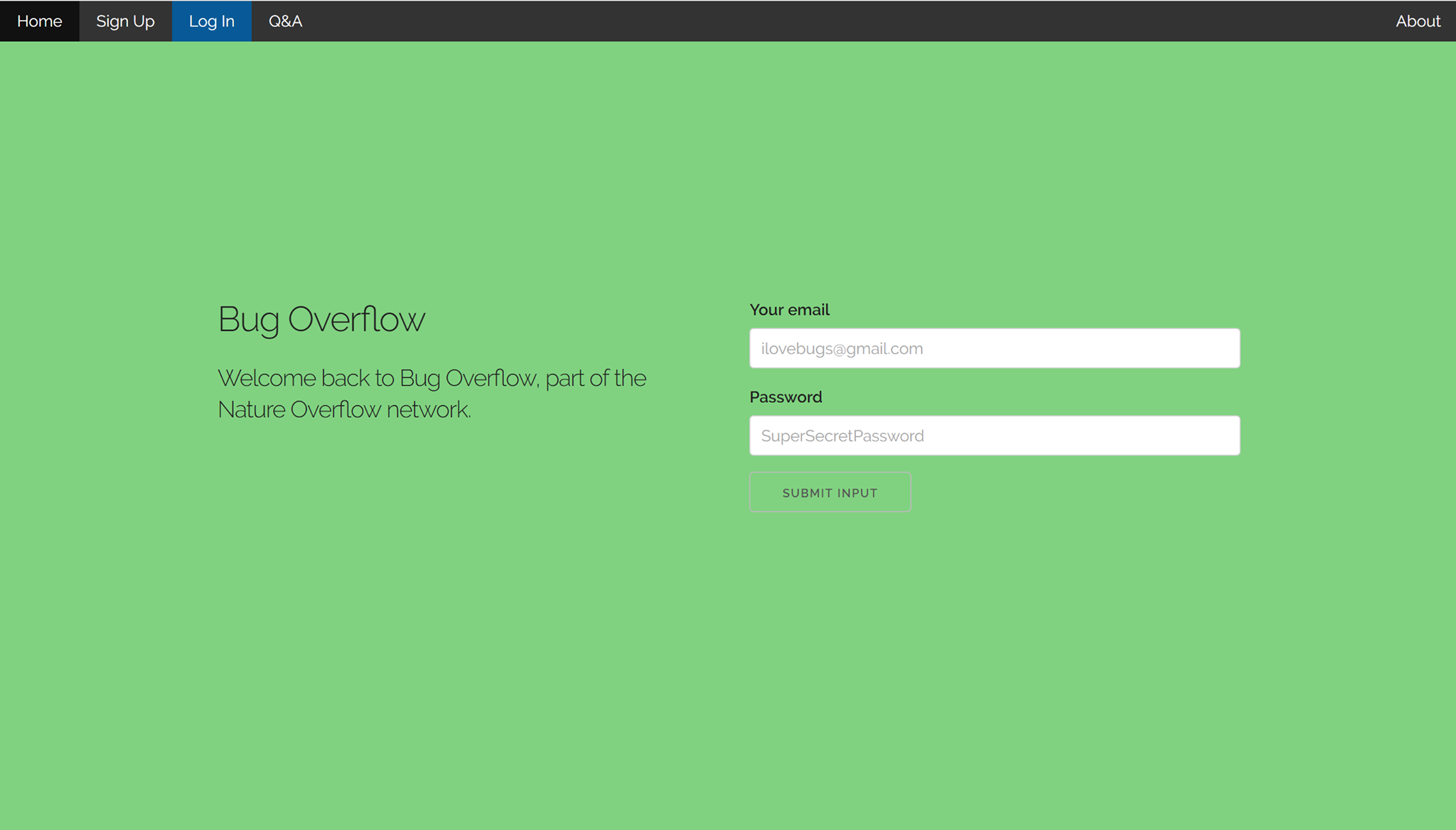


Figure 7, bugoverflow login page

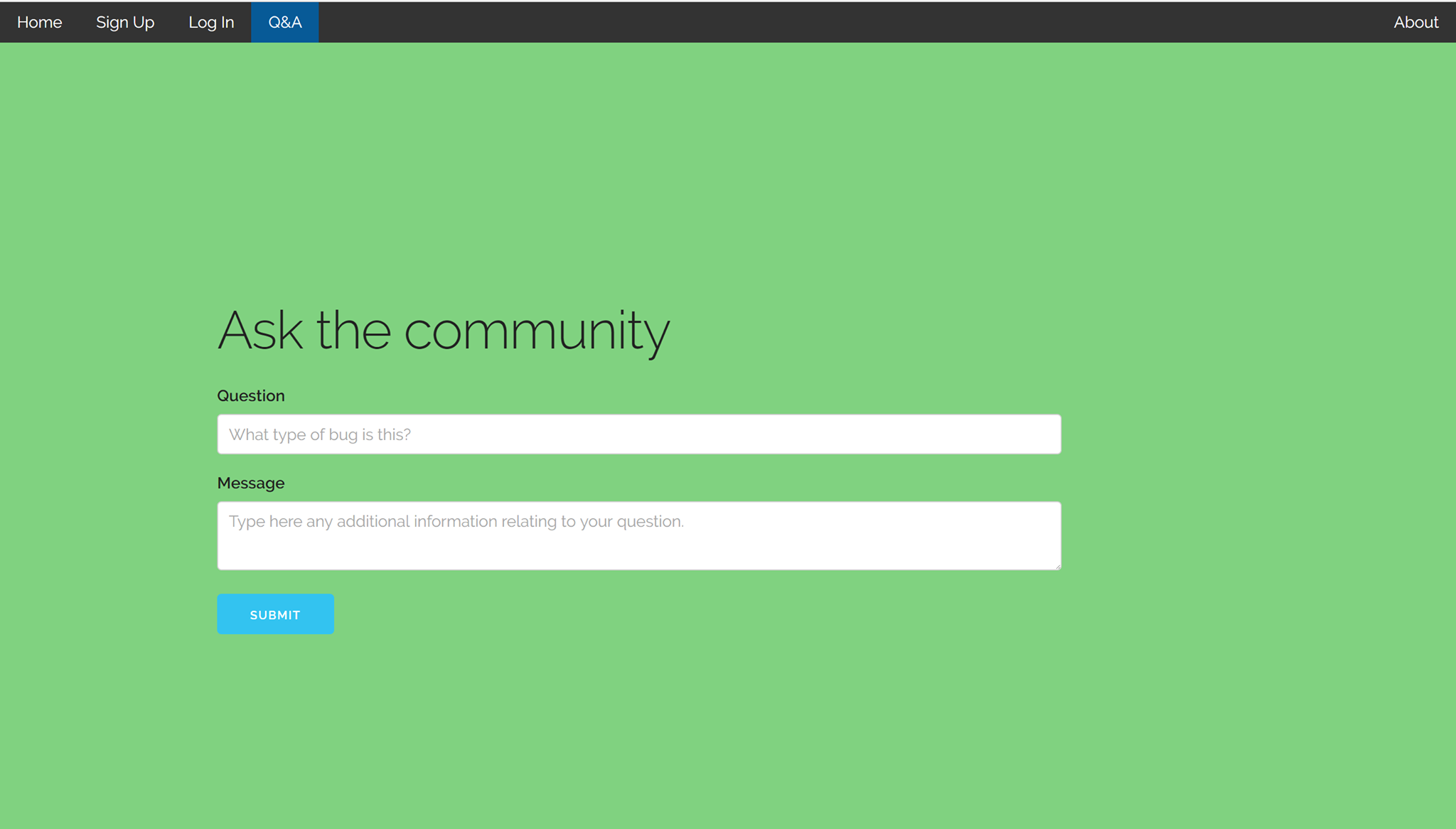


Figure 8, Q & A page still under development

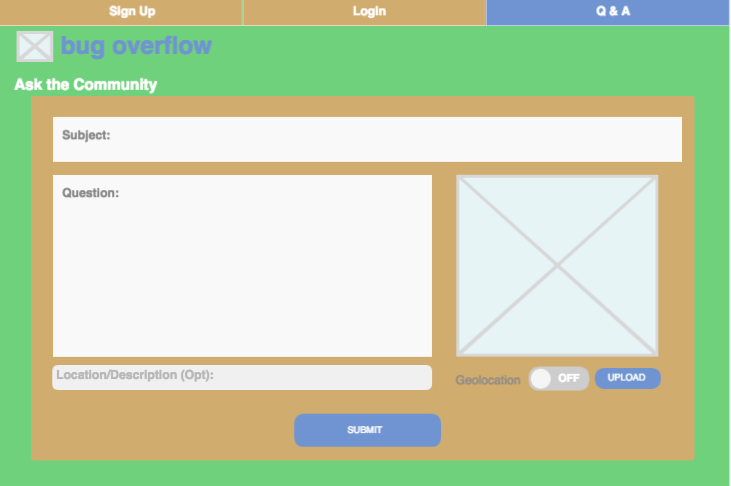


Figure 9, wireframe of Q&A page featuring image upload and geolocation properties

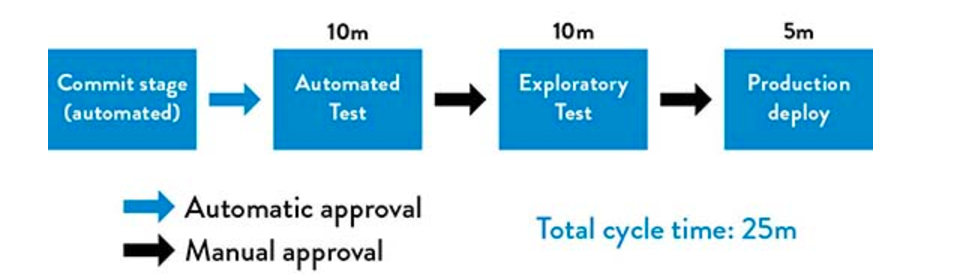


Figure 10, Google’s deployment pipeline (Source: Kim et al., The Devops Handbook, 2016.)