CottShare

Siyue Yu - c3yusiyu
Trevor Anderson – g4anders
Abdulmecid Ceylan – g3ceylan
Liam Jackson – g4jackso

Github Repo: https://github.com/Orangehop/csc309assignment5

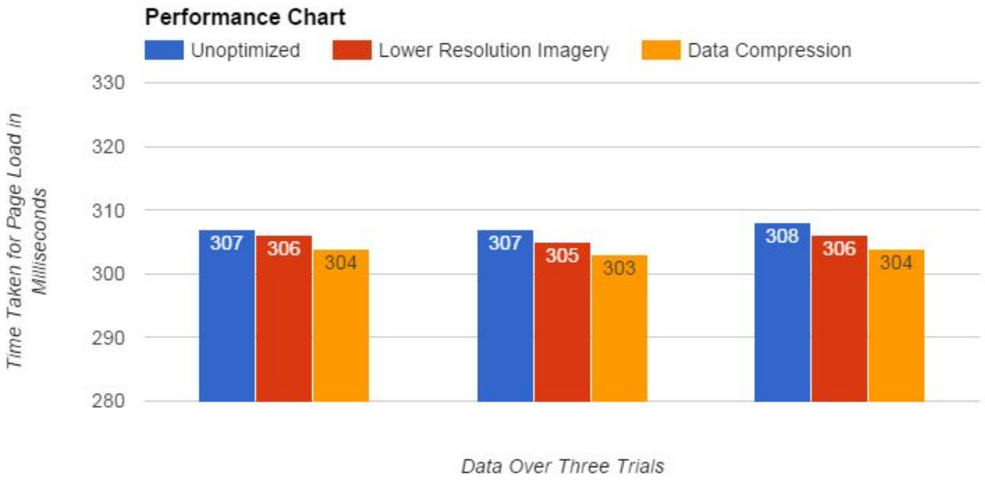
CottShare is a website built and designed with the purpose of managing an economy based around sharing cottages. Each user on the CottShare has a profile, including their name, contact information, a brief description, a rating, and a list of properties this user owns. Each user can view a list of cottages on the website sorted by different methods, and view each listing separately. Each cottage listing, which a user can create, will include a description of its property, its location, its rent, the owner of the property and their information, and a list of comments made by other users.

In terms of a high-level view of our software, we split on our software into three sections, front-end client side, back-end server side and our database. Our front end is a multi-page system of 3 files, index.html, profile.html and listing.html, and their corresponding JavaScript files, index.js, profie.js, and listing.js. Upon entering the site, you are prompted with 3 options. To log in if you have an existing account, to sign up if you would like to create an account, or to log in via Facebook if that is your preference. Choosing to sign up, you are prompted with 3 fields, your email address, your password, and a field to confirm your password. After all 3 fields are validated; you are taken to the home page, where you have the choice of what to do next. You can create a listing, where you will be asked to input all important information about the cottage, including its geolocation. You can also search for a listing. From here, you can enter in a location, and the website will display all listings within the area you searched for. You can also at any point, click on Profile or My Listings in the upper right hand side of the browser window. Clicking on Profile will take you to your profile page. On here you can view the information about yourself, as well as input any new information about yourself, such as your phone number or location, and a description about yourself. Clicking on My Listings will take you to a list of all properties that you have created. Selecting one of those will take you to a listing page for the cottage. On this page you can view and edit any of the information about your listing, such as location, price, and availability. You can also view comments made on this listing by other users who've viewed it and quite possibly rented it. To summarize, we have our entry page, which allows you to sign in or sign up, whether it be from user put in information or Facebook authentication. From here, you enter the home page, where you can search or create listings, or view your own listings or profiles. On either of those, you can edit and add any new information about yourself or your cottage. Creating a listing will allow you to input information to create a new page than anyone can view and will show up on your profile, and searching for listings will allow you to parse through all cottages in the database by their location relative to the location you've entered.

For a list of pages, we have index.html, profile.html and listing.html. For UI elements in index.html upon entering, we use bootstrap buttons to present the options to log in, sign up, or authenticate via Facebook. Clicking log in or sign up provides text fields to input information. Once logged in, there are two more buttons to search or create a listing, and a banner up top to return to the home page or to view your listings or profile. In profile, you are displayed a panel. Within this panel there is information about yourself in paragraph tags, and glyphicons for visual appeal next to them. There is a table to neatly display the properties you own, and a star rating UI to provide a rating for each user. The star rating JavaScript and css was source from http://plugins.krajee.com/star-rating and was altered slightly to fit our desires. There are a couple of buttons on each profile at first glance, one to contact this user, which will reveal a text area to write a message to them and a button to send it. There is also a button on the top of the panel to change the text fields with your information in them to text inputs to allow for the alteration of information on your profile, and the edit button is replaced with the save button to save any alterations you have made. Clicking on create a listing takes you to a page with various text inputs, to input any information about a property you'd like and a button to create the listing with that information. The button to search for listings takes you to a page with a single text input, and a button. The text input is for a user to input which location they'd like to search for listings around. The text input will assist in autocompleting cities, and other locations based on the currently entered letters. Clicking the button will parse through the database and display listings in that area. Once on a listing page, you are shown a panel, similar to the profile. On this panel, you are displayed text information about the cottage such as its location, rent and description. There is a table to display the dates this property is available and another table to showcase all comments written about this property. There is also a text are to write your own comment and a button to submit such comment to the listing.

In terms of security, we've taken measures to help prevent any sort of malicious attempt at breaking the website. We've implemented hashing using the bcrypt library to hash the passwords that any new user inputs. This prevents any hacker from simply getting a hold of our saved passwords and being able to simply look at the passwords, as hashing will make those a jumble of alphanumeric keys they won't be able to comprehend and use. We also utilize sessions to make sure that a user has logged in before they can make any rest calls that go through to the back end.

For increasing performance we utilized middleware compression, this allows the amount of data being to shrunken down when transferred then returned to its original state afterwards. We also reduced the resolution of our static images to improve performance further. Using locust.io, we generated the following data. An increase in performance of 3 milliseconds can be observed by combining these techniques.



-