Due 4/27/16	Due 4/27/16	Due 4/27/16	Due 4/27/16	Due 4/27/16
Present your team's front-end. The front-end must use an "above the fold" design. You must use flexbox to arrange content. Include media-queries in your design so that your design looks good in any viewport (browser window: screen size). The names of the members of your team should be listed somewhere on your webpage(s). Your team will field up to 10 questions about the code. You will round-robin between team members to answer the questions. If the team-member whose turn it is to answer the question cannot answer the question, the entire team loses 20% (eg, even if one team-member wrote all of the code, everybody on the team needs to understand the code and be able to explain it). Reference the repo linked here for a code sample. Bonus: earn 20% extra for your team by including at least one inline SVG that is styled with CSS (you can find code samples for SVG in my html-css repo also).	Above the fold design	Using flexbox	Using media-queries	Included team member names

Due 5/2/16	Due 5/2/16	Due 5/2/16	Due 5/2/16	Due 5/2/16	Due 5/2/16
Present your team's server side code. The server side code should serve your pages and all resources. Your website must be accessible via the web (not just localhost). Create a session for every user that comes to your website: you may use app engine's user services for this (appengine/user) or a cookie with a uuid. Your webpages should visually show that state is maintained between each request from every unique user having a session. Your team will field up to 10 questions about the code. You will round-robin between team members to answer the questions. If the team-member whose turn it is to answer the question cannot answer the question, the entire team loses 20% (eg, even if one team-member wrote all of the code, everybody on the team needs to understand the code and be able to explain it). Bonus: earn 20% extra for your team by (1) passing some value through the URL and then retrieving it and (2) explaining on a page in your website how this could be helpful in maintaining state.	Serves all pages	Serves all resources	Accessible via the web	Creates a session	Visually showing state is maintained

Due 5/4/16	Due 5/4/16	Due 5/4/16	Due 5/4/16	Due 5/4/16	Due 5/4/16	Due 5/4/16	Due 5/4/16
Present your team's server side code. The server side code should allow a user to "sign-up" for an account on your website. User names must be unique within your website. Use AJAX to let the user know if a user-name is already taken. Store user information in memcache and datastore. When you retrieve user information, try to retrieve it from memcache first. If it's not in memcache, retrieve it from datastore and also then store it in memcache. Allow a user to update their information. Your team will field up to 10 questions about the code. You will round-robin between team members to answer the questions. If the team-member whose turn it is to answer the question cannot answer the question, the entire team loses 20% (eg, even if one team-member wrote all of the code, everybody on the team needs to understand the code and be able to explain it). Bonus: earn 10% extra for your team by validating email addresses on the client-side using HTML. bonus.Bonus: earn 20% more extra and then also validating the email address on the server-side.	User can sign-up	User names are unique	Uses AJAX to notify user if a user name is taken	User info stored in datastore	User info stored in memcache	retrieves user info from memcache first, then datastore and puts into memcache	User can update their info

Due 5/9/16	Due 5/9/16	Due 5/9/16	Due 5/9/16	Due 5/9/16	Due 5/9/16
Present your team's server side code. The server side code should allow a user to upload files to google cloud storage (GCS). The user should also be able to see all of their files by name, they should be able to see their file contents in a browser, and they should also be able to download their files. You should limit the types of files that a user can upload. Your team will field up to 10 questions about the code. You will round-robin between team members to answer the questions. If the team-member whose turn it is to answer the question cannot answer the question, the entire team loses 20% (eg, even if one team-member wrote all of the code, everybody on the team needs to understand the code and be able to explain it). Bonus: earn 5% extra for your team by providing some type of processing on text files. bonus.Bonus: earn 20% extra for your team by letting a user know if another user has stored the exact same file. bonus.BonusMax: earn 25% extra for your team by processing images to be two different sizes.	User can upload files to GCS	User can see all of their files by file name	User can see file contents in browser	Use can download files	File types to upload are limited

Due 5/11/16	Due 5/11/16
Present your server side code. Integrate some external service into your website, eg, like the giphy example, or github's markdown to html api, or something else. Your team will field up to 10 questions about the code. You will round-robin between team members to answer the questions. If the team-member whose turn it is to answer the question cannot answer the question, the entire team loses 20% (eg, even if one team-member wrote all of the code, everybody on the team needs to understand the code and be able to explain it). Bonus: earn 50% extra for your team by bringing some new piece of Go code to share with the class which is integrated into your website.	external service integrated