**CSE 216 Project Manager Report: Phase 3**

List your team members, providing name, email, and role for this project. Also provide the URL where your live Heroku-hosted web front-end can be found.

**Project Manager:** Allison Codi

**Backend:** Ryan Stelly

**Admin:** Ji Ho Choi

**Web:** Tanqiu Jiang

**Android:** Karthick Sivakumar

**Heroku Web Front-End:** <https://bmw-dba.herokuapp.com/>

**Back-End Server**

**Describe the mechanism you are using to interact with Google Drive. Are you using an existing library? Are you using a REST API? Are you using some other mechanism? Were you able to get the service account to work correctly?**

Ryan followed an online tutorial that detailed how to interact with Google Drive. He attempted to use the Google API Java library to do this. He was not able to get it to work successfully in his post route. When he hard coded the upload/download functionality into his main method, he was able to successfully upload/download files to the Drive, however he received a 500 Internal Server error when attempting to do this in his post route.

**Explain how you are using memcachier to cache documents.**

The memcachier stores files for 24 hours. If a user wants to repeatedly view a document, the application will not need to reload it every time within the 24 hour period, making the application more efficient.

**Is the back-end code appropriately organized into files / classes / packages?**

Yes

**Are the dependencies in the pom.xml file appropriate? Were there any unexpected dependencies added to the program?**

Yes, the dependencies in the pom.xml file are appropriate. There were no unexpected dependencies. Ryan only added memcachier and Google Drive dependencies which were to be expected.

**What was the biggest issue that came up in code review of the back-end server?**

The main post route would not work when the upload functionality was added to it. Originally, this was designed to be a separate route, but due to complications it was added to the main route for posting a message to the database.

**What technical debt do you see in the current back-end server implementation?**

In the next phase, we will need to get the post route to work for uploading a document rather than hardcoding it into the main method. Additionally, messages with likes cannot be deleted currently for unknown reasons. The userId error from phase 2 was never refactored, so the next backend will need to fix this technical debt as well.

**Describe any refactoring that was performed to reduce technical debt from the last phase.**

Refactoring from last phase was not able to be performed successfully. The userID is still hardcoded to be 1.

**Web Front-End**

**Describe your solution for file upload.**

Files were written and attempted to be sent to the backend in the form of a byte array. Unfortunately, the main backend post route did not work successfully so the frontend was not able to upload files successfully.

**Describe your solution for accessing images, documents, and links.**

Accessing images, documents, and links are all tied to the main route in the backend that was not working. Due to this, web was not able to successfully access these images, documents, and links.

**Is the web front-end appropriately organized into files / classes / packages?**

Yes

**Are the dependencies in the package.json file appropriate? Were there any unexpected dependencies added to the program?**

The dependencies in package.json are appropriate. However, there were two files called deploy.sh and ldeploy.sh that conflicted with each other. They were not the same and caused problems when attempting to deploy the code. Tanqiu edited ldeploy.sh to match deploy.sh in order to fix these dependencies.

**What was the biggest issue that came up in code review of the web front-end?**

Web frontend had limited functionality due to dependencies on backend that were failing. The largest issue was the dependency on the main route for posting a message.

**What technical debt do you see in the current web front-end implementation?**

The failing profile page from last phase was refactored this phase. There is still technical debt with encrypting messages being sent to frontend, in addition to the comments being displayed in a table. There is additional technical debt of accessing images/pdfs due to the backend route not being completed.

**Android App**

**Which approach to the Camera API did you use, and how well did it work?**

An Android built in Camera API was utilized that worked well. It was utilized within an OnClick listener, so the API was pulled up if a user clicked a button saying they wanted to take a picture.

**Describe your solution for accessing images, documents, and links.**

Glide was used for accessing images from the JSON file and displaying them into an ImageView. Glide also has a cache built in so images do not have to havebe reloaded each time.

**Is the android app appropriately organized into files / classes / packages?**

Yes

**Are the dependencies in the build.gradle file appropriate? Were there any unexpected dependencies added to the program?**

Yes, the dependencies in the build.gradle file are appropriate. There were not unexpected dependencies in the program.

**What was the biggest issue that came up in code review of the android app?**

There was an issue with the Android app not being able to connect to the database after a change was made to the backend. This prevented progress from being made in other parts of the application.

**What technical debt do you see in the current android app implementation?**

The OAuth client ID/token issues from the previous phase still need to be refactored. Additionally, when an image is added to a post, it automatically posts rather than giving the user the option to go back and add a message first. Can’t send the image with text, posts automatically instead of going back to message

**The Admin App**

**Describe your solution for upgrading the command-line interface for administrative tasks.**

The CLI was updated to have a different main menu allowing for the administrator to choose if they’d like to manage the content in the Google

Drive or the content in the database. As specified in the rubric, the CLI can manage storage in the Drive and delete files that are older than a month old as the cache starts to fill. Inappropriate documents can also be removed from the drive.

**Describe your policy for deciding when to delete files.**

Files can be deleted by date posted (older than a month), post number, or file name. There is an option for the admin to list all of the files and choose which one they want to delete if it is deemed inappropriate.

**What was the biggest issue that came up in code review of the admin app.**

Ji faced issues connecting to the Google Drive and had to figure out how to authorize admin to make delete requests to the drive. There continues to be an issue with deleting a post when it has votes on it.

**What other roles did the admin developer play in helping the project to move forward toward completion?**

Ji made the CLI much easier to interact with by reworking the main menu. The storage remaining in the cache can be displayed, and posts that are over a month old can be deleted to make more room in the cache.

**Project-Wide**

**Were there any team issues that arose?**

The team faced dependency issues on backend. Frontend struggled on parts of their code due to these dependencies. However, the team had a positive, friendly team dynamic to try to work out the issues we faced. Although the phase was not completely finished, the team worked together to get as much done as possible.

**How did the amount of time your teammates spent compare to the amount of time you thought the tasks would take?**

Backend took a lot longer than anticipated and was not able to finish. Ryan worked hard and put a lot of effort in but faced some roadblocks he was not able to solve by the deadline. The memcachier worked, but the route for uploading images and files did not. Frontend took longer than anticipated due to dependencies on the post route for images/files, so they were not able to finish completely either.

**Describe the most significant obstacle or difficulty your team faced.**

The team faced issues due to the main post route in backend failing. Originally, we planned on having separate routes for posting/getting files, but backend had a hard time implementing this. Backend switched to make posting an image/file part of the original route to post a message to the database. This caused unexpected problems for frontend, and they had to change their code to use the updated route. The post route never worked fully, so frontend could not test their code the way they would have liked to.

**What is your biggest concern as you think ahead to the next phase of the project?**

Our team has accrued a lot of technical debt that must be fixed. We are hoping that this debt does not conflict with what we plan on doing for Phase 4, but if it does we may run into more dependency issues that we need to combat.

**What are the features to be implemented in Phase 4.**

We plan on implementing a search bar to the application. This will allow users to search for keywords in a post rather than scrolling through all of the messages to find what they are looking for.

Submit documentation for the user interface for the apps.

**![A screenshot of a video game

Description automatically generated]()Android UI:**

**![A screenshot of a video game

Description automatically generated]()![A screen shot of a computer

Description automatically generated]()**

🡨Updated UI for posting a message. Now contains ability to open camera and open files to add to the post.

🡨 Camera API UI

Gallery API UI 🡪

**Admin UI:**

**![A screenshot of a cell phone

Description automatically generated]()**

**![A screenshot of a cell phone

Description automatically generated]()**

^^ Display of all files in database and delete based on ID number

🡨 Main Menu with added options of content management or database management

**Web UI**

Not pictured. Issues deploying due to backend dependencies. Attempted to add functionality to display images and add files to posts. Profile functionality now fully working (technical debt from last phase).