**Part 1**

I have made repositories in the past—just look through my GitHub page and you’ll see.

https://github.com/RemnantKnight56

Here is the personal repository for this class:

<https://github.com/RemnantKnight56/CSE-490R-Personal-Folder>

I didn’t make any specific notes last week or this week—I wasn’t aware I needed to, but I’ll make notes for meetings in the future.

**Part 2**

Github is based the Elements of Software Configuration Management

* **SCI – Software Configuration Item** – Software that is being changed
* **ECP – Engineering Change Proposal** – Requirement or request for change
* **SCCB – Software Configuration Control Board –** People who approve or deny proposed changes.
* **PSL – Program Support Libraries –** The location where all software reside—such as the cloud for GitHub
* **DO – Design Object in Development Domain** – Version Control – The whole GitHub software
* **CCC – Change & Configuration Control Environment** – The software that changes are being made with – IDEs

Usually start with a bug report, enhancement, or change requirement. Become a proper report or request for change. Gets approved by SCCB and gets put into backlog. Can take the form of a story, deficiency, etc. The change is made by an engineer and put into existing codebase. All changes are recorded in library, as well as software artifacts. Software artifacts are especially important to quickly identify changes in case something breaks. Once change is complete, the change is reviewed, future items are brought up in meetings and assigned to engineers. Tests are conducted later, and if anything is found, a bug report is created, and the cycle begins again.

For version control, the proper method is to take the current version of code and make your own branch to make changes from, then commit the change. Merge the change afterwards, but if there are conflicts, update the code, then merge and commit again. These changes are made back to the version layer, but if this takes too long, it can end up in an older version. The updates to that older version need to be carried over to your change, then merged into the latest version.

If there are two baselines, they will need to eventually be merged, but keep track until then.

**Part 3**

Again, no notes were taken in meetings, but I did accomplish my part of things—I assisted in making one of the three Readmes that will be part of the project, as well as providing a pair of stories for the Tasks and Accounts portion of the requirements.

Jeremy Cole

Bi-Weekly Status Report 2

**Overall Status:** On Schedule

**Summary of Project Tasking:**

Attended all meetings save for one last week, due to my work schedule preventing me from joining. I was assigned to work on the Tasks and Accounts stories and completed two yesterday.

**Number of Hours:**

Including meeting times, roughly 12 hours have been worked these past two weeks, but we are still in the early part of the semester given we are still establishing workflows. That should amount to about 21 hours so far.

**Accomplishments:**

Number of Story Points: 2

Number of Stories: 2

Personal Velocity: Good

Project Velocity: Good

Earned Value: 1?

**Performance:** Last week was slow, but we needed to be given requirements to work with for stories, and that did not come around until Thursday this week.

**Challenges:**

The challenge this week was having patience to wait for others to give us—the development tools team—what we needed to finish, as well as testing the layout of the readme file to make sure no confusion was had.

**Plans:** Actually get started with development, ideally, as well as expanding on my personal folder. Until there are stories to work on, these reports can only be so detailed.