Logistics

Work Room :

- Jack Baskin Room 316 - main workroom

Meetings :

- Monday- JBE Rm 316 - 4:00-4:30 (T.A/ Daily Scrum Meeting(15min))
- Monday- MEP- 4:30-5:30 (Sprint Planning Meeting)
- Wednesday- Northrop- 6:00-7:00 (Daily Scrum(15min))
- Thursday- Northrop- 7pm-8pm (Daily Scrum(15min))
- Sunday- Virtual-Time- Sprint Review and Retrospective

Project Repository:

https://github.com/futureistaken/MovieNight

- We should create a folder in our master repo that contains all of our sprint plans/reports. So that it is not cluttered.
- Each feature should be implemented within a branch then when the feature is tested and working we should merge it into the master branch (continuous integration)

Communication Channels:

- The main way of communication is through text within a group chat
- Virtual Meetings made through Google
- Put all Sprint plans/review/release material into google drive shared CSE 115 folder

Team Responsibilities:

- Be mindful of everyone's time
  - If can't make a meeting, let everyone know beforehand
  - Be responsible in knowing when meetings are supposed to happen to allocate time for them within your schedule (sometimes times can be adjusted but first, communicate with the team to let them know you cannot make a specific time)
  - Set meeting times if not set already a day in advance preferably or at least 6 hours before
  - Remember ~~EVERYONE~~ everyone (no need for caps) is busy

Development Environment

Platform:

- Anna - macOs
- Justine-
- Joe -
- Babak- macOs

Tools we working with…

- AWS - deployment
- Languages
    - Python
    - SQL
    - JS
    - HTML
    - CSS
- Frameworks/libraries
    - Flask
    - Bootstrap
    - jQuery
- Toolkits
    - SqlAlchemy- connection between sqlite/flask
- Database
    - Sqlite
        - Table 1 - users
        - Table 2 -movies connected to API

IDE:

- Anna- VS Code
- Justine- VS Code
- Joe-
- Babak- Pycharm

<u>Work Patterns</u>

Definition of Done:

- **<u>Before merging into master:</u>**
    - Define unit tests (what do we need to test for with this feature)
    - Run unit tests (make sure no errors or warnings)
    - Make sure code is clean/commented where needed

- **<u>After Merge:</u>**
    - Integrate into the master project
    - Run to make sure added code creates no conflicts nor breaks the project

- **<u>Sprint Task completion:</u>**
    - Task fully implemented/completed
    - Doesn't have to be committed to master but should be committed and pushed to branch
    - Following guidelines of before and after merge
        - Hence run all tests/integration guidelines

        - Mark task as done on asana

- **<u>User Story completion:</u>**
    - Code merged into master and integrated properly following before and after merge guidelines
        - Run tests/integration guidelines
    - All sprint tasks for user stories complete and fully integrated within master
    - Mark user story as done on asana
    - Let scrum master know user story is complete so that burn up chart can be updated

- **<u>Suggestions from slides we could possibly use</u>**
    - Code reviewed for standards compliance
    - Code reviewed by a team member or Walk-through performed
    - External/Public API documented
    - Non-functional tests (e.g. usability, performance) passed Regression tests run without error Static code analysis performed and passed
    - Test coverage measured and achieved

Acceptance Criteria:

- Proper user story implemented, code does what we wanted it to do on a higher level
- Run test to make sure feature is fully implemented and does what we need it to do
- Every user story should have acceptance criteria
    - Is this what we wanted?
        - What do we want?
            - Set guidelines
    - Does it pass all the tests?
        - Guideline tests it should pass

Technical Practices we should/could incorporate:

- Done Criteria
    - Follow guideline criteria for each push, task, user story
- Peer review / Pair Programming
- Clean code
    - Comment code when needed
    - Use good variable/function names
    - Reuse code (write functions don't repeat)
- TFD (test-first development)/TDD (test-driven development)
    - Unit test all features before integration
    - Test after integration as well
- Continuous integration
    - Push to github after tasks complete

    - Merge to master if other teams members are relying on your code especially after your feature is done
    - Push all code onto master after user story complete
- Version Control
    - Use Github
    - Use branches
    - Communicate during merge conflicts if unsure
- Test coverage criteria
- Static Analysis Tools

Clean code guidelines:

- Write simply
- Properly indent
- Comment when needed
- Good naming conventions
- Style guide
    - https://legacy.python.org/dev/peps/pep-0008/
    - http://google.github.io/styleguide/pyguide.html
- Avoid duplication
- Explanatory variables
- No magic numbers
- Encapsulate conditionals
- Avoid negative conditionals
- Encapsulate boundary conditions
- Functions do one thing