Table 1: Revision History

| Date | Developer(s) | Change |
|---|---|---|
| September 26 | N/A | Initial documentation |

# Development Plan
# Mechatronics Engineering

Team #1, Back End Developers
Jessica Bae
Oliver Foote
Jonathan Hai
Anish Rangarajan
Nish Shah
Labeeb Zaker

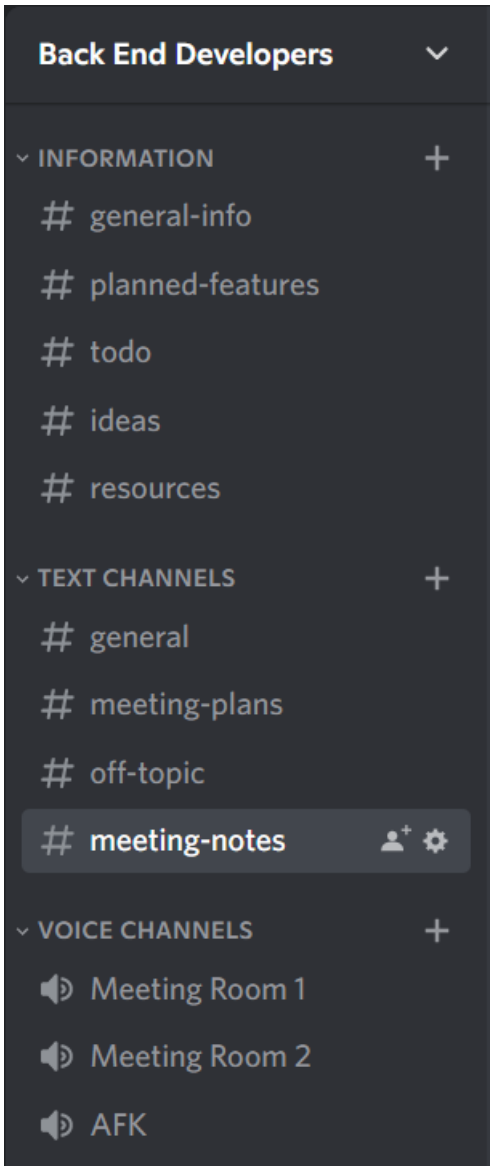
## 1 Team Meeting Plan

Weekly meetings are to be held every Saturday at 8:00PM, and additional meetings are to be planned as needed. Any meetings requiring the attendance of the team's supervisor will have to be planned based on Dr. Macedo's availability. In cases where some team members are absent, each absent member is expected to bring forward their discussion topics prior to the scheduled meeting time, and review the meeting notes posted in the Discord server. Note-taker is expected to write down meeting notes for each meeting, regardless of the number of participating members.

# 2 Team Communication Plan

All primary communication is to be done through the team's Discord server, including all meetings. Secondary communication will be through the Facebook group chat only if necessary.

The following is a screen shot of this team's Discord server:



# 3 Team Member Roles

| Role | Name | Description |
|---|---|---|
| Team Leader | Jonathan Hai & Jessica Bae | Organize meetings, assign specific tasks to all team members, overlook deliverable deadlines, keep track of overall team project progress, communicate with the TA, the professor, and the supervisor |
| GitHub Leader | Labeeb Zaker & Nish Shah | Review and approve all Git merge requests from the team. |
| Meeting Coordinator | Anish Rangarajan | Lead the conversation for team meetings. Responsible for preparing meeting agenda. |
| Note-Taker | Oliver Foote | Take notes during the meeting and post it in the meeting-notes section of Discord channel for record keeping purposes. |
| Embedded Developer, Frontend Developer, Backend Developer, CAD Designer, Electromechanical Designer | Everyone | Everyone will work together on the technical component of this project. |

The task of reviewing rubrics, and each subcomponents of the project will not be assigned to any specific members. These tasks are to be allocated to different members each time as and when needed.

# 4 Workflow Plan

1. Move current task into the To-Do Section of the Trello Board. Assign relevant labels through the Trello Card (Hardware, Software, Features, Bugs, etc.).
2. Make branch for current task on Github (Branch named according to: ”Trello Ticket # - Member Name”).
3. Once task has been completed, move task into Testing Section of the Trello Board.
4. If testing is successful, send pull request and move task into the Code Review Section of the Trello Board.
5. Update code with any comments from other developers.
6. Once the pull request has been approved, move the task into the Finished Section of the Trello Board.
7. Inform team that the branch has been merged into the main branch.
8. Delete your extant branch.

# 5 Proof of Concept Demonstration Plan

The risks for the success of the project lie primarily within the software that is designed. The hardware component in which data is collected will be designed with a relatively simplistic plug-and-play approach with the goal of feeding data to the software component of the project. Several off-the-shelf sensors will be used to reduce the development overhead and the many possible problems that developing a complex electromechanic system can involve. However, a proof-of-concept demonstration is still necessary regarding the ergonomics of the project. Should the hardware take on the form of wearable technology (as is currently planned), a demonstration is necessary to show that the system is possible to be worn comfortably, safely, and unintrusively during the user's daily activities.

The software system will be the lynchpin of the project. Its goal will be to take the various events detected by the hardware system that are relevant to Ecological Momentary Assessment (EMA) and handle them accordingly. This will involve logging timestamps, locations, environmental factors, and other momentary information, and prompting the user to answer self-survey questions relevant to their current situation. The software system will then process the answers to these questions and the momentary data. Finally it will send off the data to the team of physicians handling the user's EMA at the end of the monitoring period.

Other than the ergonomics, if the hardware component of the system is incapable of sending data and handling output from the software, continuing with the project involves finding a new method of data collection and transfer. If the software component of the system

is incapable of responding to data inputs from the hardware, processing said data, and returning meaningful results, then the goal of the entire project is rendered moot.

The goal for the proof of concept demo will be to demonstrate a functional EMA-enabled software system that can respond to events relevant to EMA, and respond accordingly. This software system will be run locally on a team member's computer. It will be capable of:

- Accepting inputs in the form of momentary data from events that are triggered as a simulation of real EMA events (such as limps, falls, strange movement patterns, etc.)
- Prompting the user to answer relevant self-survey questions.
- Displaying information and intaking information from the user simplistically according to common HCI guidelines.
- Processing the inbound EMA and survey data and producing results meaningful to the physicians responsible for the user's EMA.
- Producing graphical representations of said EMA data.
- Sending the processed data and representations to the physicians responsible for the user's EMA.
- Notifying the user about relevant information as a result of processing data (confirmations that data has been sent, recommendations for activity or rest, etc.)

Regarding the hardware component of the project, it will be:

- Useable/wearable in a safe, comfortable, and non-intrusive manner optimized for human well-being and overall system performance.
- Placed in a position to collect data relevant to EMA display data regarding EMA related activities.

# 6 Technology

The following tools shall be used for development in software, embedded systems as well as any support platforms in the system:

1. Programming languages:
    - Python will be used to generate computer vision code along with several graphical and UI elements. If necessary, Python will also be used for Machine Learning integration.

- C will be used to program the embedded system (micro-controllers) for efficiency and memory constraints.
- SQL will be used for DBMS and storing data for activity-based tracking.
- R will be used for statistical computing and data analysis.

2. Platforms for development:

   - VS code will be used for code-based development.
   - Arduino IDE may be used to code for several existing sensor libraries (gyro meters, Bluetooth modules, Wi-fi modules, etc).
   - STM32Cube IDE for STM32 development.
   - Autodesk Inventor may be used for creating models for the device and finishing product.
   - MySQL will be used as the platform for interacting with the Database.
   - The R environment will be used for data manipulation, calculation and graphical display

3. Version control platforms:

   - GitHub will be used for version control for development purposes and general group activity tracking.
   - GitLab will be used for version control (collaborative) with McMaster University.

4. Document generation:

   - Latex will be used for generating documentation. TeX distributions that will be used amongst the collaborators are Texmaker, TeXworks, MikteX and VScode Latex extension.

5. Specific plans for Continuous Integration (CI)

   - CI will be used to block merge requests until all tests have been passed and approval has been given by atleast two members of the team.
   - This will speed up development by reducing the overall bugs introduced into the project.

6. Measuring tools for code:

   - Coverage.py for measuring code coverage in Python and effectiveness of testing.
   - Valgrind for efficient memory management.
   - Bullseye coverage for C/C++.

7. Libraries to use:
    - OpenCV: Open Source computer vision library capable of performing image and video manipulation.
    - Numpy, pandas: Libraries with Python for number and matrix computation/manipulation.
    - HAL libraries: Should we pursue using an ARM architecture chip, the HAL libraries will allow for ease of coding.
    - tkinter: Default python UI library.
    - Several Arduino Sensor libraries to use for interacting with accelerators and other wireless modules.
8. Tools to use for project:
    - 3D printer and slicer tools for rapid prototyping.
    - Soldering station and Glue Gun.
    - Wood working tools (Hand Saws and Power Sanding ) if necessary.

# 7 Limitations of Tools and Methods of Solving

- Python: Memory usage may be inefficient due to limitations of hardware.
  Solution: Use C or utilize static memory allocation.
- Arduino IDE: Libraries may be specific to the Arduino Microcontrollers.
  Solution: Write drivers for other microcontrollers.
- CI/CD/Code Measuring Tools: Complex to set up, may require extra learning overhead to reach full functionality.
  Solution: Assign members to investigate and document CI/CD methodology.
- Libraries: Compatibility of libraries are version dependent and not necessarily backwards or forward compatible.
  Solution: Run compatibility testing before using or changing libraries and keep updated documentation available.
- 3D Printing: High time constraints, low availability, and print quality may be inconsistent.
  Must take time constraints into consideration before integrating 3D Printing into the project.

# 8 Coding Standard

1. Python:
    - Code development will closely follow the Google Python Style Guide as a list for do's and don'ts for Python programs.
    - For inline code documentation, Google style doc-strings will be followed for commenting.
    - Pylint will be used as a linter for code development. This linter can be used as an extension to VS code while developing and debugging code.
2. C: lint will be used as a linter when programming in C for development and debugging purposes.

# 9 Project Scheduling

1. Master Project Schedule will be made with a list of all the deadlines, deliverables and Project Implementation tasks. Work will be done based on weekly sprints where tasks will be assigned to each member along with an estimated number of days needed to finish the task. This number is evaluated based on relative complexity of each task.

2. Imposed deadline for all tasks are **two** days before the due date of the deliverable. On this day, the team will conduct a review of all the work done by the members and provide any feedback. The last day before the due date is reserved for making any final changes and small revisions. Moreover, the team will check the completed deliverable with the marking rubric.

3. Every two weeks, a design overhaul will be conducted wherein the team will go through the previous parts of the project and update it with any changes made as a result of the current iteration of the project.