Table 1: Revision History

Date	Developer(s)	Change
2022-09-26	Jonathan Cels	Team meeting and communication plans, personal role, workflow plan, coding standard, project scheduling, started technology section
2022-09-26	Alexander Van Kralingen	Introduce CI/CD plan, team member role
2022-09-26	Arshdeep Aujla	Introduce Hardware List 6, 3.2 Add Roles
2022-09-26	Joshua Chapman	Modify Hardware List 6, 3.2 Add Role
2022-09-26	Alexander Van Kralingen	Fix author field not being populated bug
2022-09-26	Alexander Van Kralingen	Populate proof of concept plan
2022-09-26	Rupinder Nagra	editing

Development Plan
Chess Connect

Team #4,
Alexander Van Kralingen, vankraa
Arshdeep Aujla, aujlaa4
Jonathan Cels, celsj
Joshua Chapman, chapmj3
Rupinder Nagra, nagrar5

2022-09-26

1 Team Meeting Plan

The team will meet weekly on Thursdays at 10:20 AM until 11:20 AM. Team members are expected to attend the Thursday lecture and meet up following the lecture. On the event that no lecture is scheduled, the meeting time shall be changed to 9:30 AM until 11:20 AM.

Additional meetings will be scheduled when necessary using the communication methods outlined in the Team Communication Plan.

2 Team Communication Plan

The team will communicate over a Discord group channel. Each team member is expected to have Discord downloaded and readily accessible on at least one of their devices. In the case of emergencies or time-critical situations, a team member will use SMS text messaging in place of Discord. Text messages will only be used in critical situations.

Issues will be managed using a Kanban board, implemented with a Git project board. Tasks will be split into small, workable items with differing deadlines and importance.

3 Team Member Roles

3.1 Alexander Van Kralingen

- Embedded systems software development

- Continuous Integration/Deployment management
- Assisting with development of web application
- Code reviewing for both embedded and web application code
- Unit/integration testing development

3.2 Arshdeep Aujla

- Assisting in building the chess board and customise the chess pieces
- Installing the hardware components on a breadboard
- Soldering components on the PCB
- Mapping the relation between the inputs and outputs using Karnaugh Map and State Machine Table

3.3 Jonathan Cels

- Enabling the Bluetooth connection between the application and the microcontroller
- Assisting with development of web application
- Programming the microcontroller with multiplexing, Bluetooth and rules of chess
- Leading testing initiatives and ensuring that thorough testing is completed
- Editing and formatting of all documentation before submission

3.4 Joshua Chapman

- Circuit design of power and transmission of sensing units
- Hardware component selection and integration
- Assisting with soldering components and mechanical assembly
- Programming the microcontroller with multiplexing, Bluetooth and rules of chess
- Team leader for purchasing and planning phase

3.5 Rupinder Nagra

- Lead front-end developer
- Deployment and testing of web application
- Back-end development for server/database management
- Code reviewing for all web application-related changes

4 Workflow Plan

The project will use the feature-branch methodology in Git. An outline of the workflow is as follows:

1. Pull any changes from the master branch
2. Create a new branch for development of a specific feature or subsystem
   (a) Use branch names that are descriptive to the feature
3. Commit code frequently with descriptive messages
4. Add unit and integration tests for the changes
5. Push code to branch
6. Create a pull request
7. Another team member reviews and approves or rejects the pull request
   (a) The tests are reviewed and more tests are created if necessary
   (b) The pull request cannot be approved if all tests do not pass
8. Merge feature branch into the main branch

5 Proof of Concept Demonstration Plan

The proof of concept demonstration will involve both a hardware component and a software component.

Hardware

- Components Involved:
  - Three different chess pieces with embedded magnets (ex: white queen, white pawn, black knight).
  - Two 3D-printed squares with Hall-effect sensors and LEDs attached.
  - LCD screen is configured and wired to display piece recognition.
- Demonstration:
  1. Separate pieces are placed on the two squares.
  2. One of the pieces will capture the other as per the rules of chess.
  3. The LCD screen will display the states of each square including the colour and type of piece.
- Risks:

1. The configuration of the sensors might be challenging to coordinate (sixty-four different sensors means many IOs to process at once). Correctly configuring two pieces means this can be easily scaled to sixty-four.
2. The sensor accuracy is something that could create a barrier to identifying six different pieces in each colour. Inaccuracy could mean misreading the identity of the piece on the square. Correctly sensing three different pieces will prove that this barrier can be overcome.
3. Supply-chain issues could mean some of the products selected for this project may not arrive on time for the demonstration.

Software

- Concepts to Demonstrate:
  - Front-end User Interface (UI)
- Demo Platform:
  - Figma Design Platform
- Demonstration
  - The UI layout will be shown, as well as all of the various components that will (eventually) connect to the back-end and hardware. The Bluetooth connection will provide the information needed to populate the UI with pieces, valid moves, best moves, game modes, etc. The Bluetooth connection will not be a part of this proof of concept because it is a tool that has been tested and proven in many other applications.
- Risks:
  1. Hardware-software connection could be problematic due to signal loss or dropped communication packets. This will not be demonstrated in the proof of concept plan. However, Bluetooth is a commonly used communication method and should be a reliable information stream for the web application.
  2. Communication speed could limit the processing of valid moves and add delays to the connection between the moving pieces and the display on the UI. The speed in which pieces are identified and processed can show that this information can be sent over Bluetooth and processed for the web app quickly and efficiently.
  3. Chess engines could be resource intensive and the web application hosting platform and user system could prevent the user experience from being smooth and responsive. Displaying a light interface without too many calculations involved will minimize the computing power needed to provide a smooth user experience.

6 Technology

- Languages and Frameworks
  - **JavaScript, HTML, CSS, and React:** Front-end web based development
  - **Python, FastAPI, Node.js:** Back-end development and Bluetooth connection
  - **C:** Microcontroller programming
  - **MongoDB, MySQL:** Database solution
- Linting
  - **ESLint:** JavaScript development in VSCode
  - **Flake8:** Python development in VSCode
- Testing Frameworks
  - **React Testing Library:** JavaScript testing framework for React
  - **PyTest:** Python testing framework
- **Heroku:** Deployment
- **VSCode:** Code editor
- **GitHub:** Version control and project management
- **LaTeX:** Documentation
- Libraries and API's
  - **Bluez, PyBluez:** Enabling Bluetooth data transfer
  - **Stockfish.js:** Chess engine to find optimal moves
- **Continuous Integration (CI)** GitHub Actions are to be used for CI, with the following general workflow:
  - Running on ubuntu-latest
  - **Pull Requests (PR):**
    - Requires one reviewer
    - Requires development branch to be up-to-date with main before approval
    - Requires building/linting and tests to run completely without errors
  - **Build Embedded Code:**
    - Triggered on PR from any hardware branches
    - Checkout hardware branch

        * Build all C code
        * Run all C tests
    - **Build Web App Code:**
        * Triggered on PR from any webapp branches
        * Checkout webapp branch
        * Lint all Python code
        * Lint all JavaScript code
        * Build JavaScript code
        * Run all JavaScript tests
        * Run all Python tests
    - **Build LaTeX docs:**
        * Triggered on push (any branch) with *.tex file changes
        * Build all latex documents
    - **Embedded Testing:**
        * Triggered on PR with Hardware Label opened, updated or closed
        * Run all C tests
    - **Web-App Testing:**
        * Triggered on PR with Webapp Label opened, updated or closed
        * Run all JavaScript tests
        * Run all Python tests

- **Continuous Deployment (CD)** GitHub integration in Heroku to be used for continuous deployment:
    - **Deploy:**
        * Triggered on push to main
        * Push to main event occurs when merging a PR
        * Web application deployed to Heroku on successfully merged PR.

- **Hardware**
    - **Design Tools**
        * Altium for circuit design and testing
        * AutoCAD for mechanical modeling and printing
    - **Chess Board:**
        * Custom chess board with embedded circuit
        * Chess pieces with embedded magnets to facilitate piece detection
    - **Electrical Components**
        * Hall sensor for piece detection
        * RGB LEDs for move display

        * LCD display on board for best engine move suggestions
        * Mechanical switch component
    - **Processor:**
        * Programmed for I/O handling and multiplexing
        * Bluetooth unit to transmit custom packets back and forth

7 Coding Standard

The project will follow the Airbnb style guide for JavaScript development, and use the Flake8 style guide for Python development.

8 Project Scheduling

The project will use a GitHub project board to track and schedule tasks on a weekly basis.

Technical roles are decided on the basis of prior knowledge and interest. In the case of the team leader, the role will change with every deliverable. Team members are responsible for decomposing their tasks into kanban items, individually or alongside other team members working on the same task.

Major milestones will be tracked on the project board. Milestones include both hard and soft deadlines for task completion. Hard deadlines are the project deliverable due dates. Soft deadlines are decided by the team for the completion of technical tasks.