1. Introduction

The *Teaching Tasks* project is a mobile application intended to be used by teachers of and parents of children with Autism Spectrum Disorder. The software provides a selection of games that the child can play while closely following the technique of **Discrete Trial Instruction** (DTI) to help them learn material set forth by the Maine Early Childhood Learning Guidelines.

1.1 Definitions

DTI: A technique found as part of the Applied Behavioral Analysis (ABA) techniques that is a method for helping children improve physical ability, cognition, social skills, speech, and independence. The methods are usually tailored towards the specific child who is being taught, but the processes remain the same.

1.2 Hardware Requirements

The software will need access to the hardware's camera for the use of profile pictures.

2. Management

2.1 Organization

2.1.1 Version Management

Github shall be used for version control. Programmers will create a branch for each feature and code specifically for that feature in order to not cause discrepancies between other branches.

2.1.2 System Building

The system requires the library "MPAndroidChart".

2.1.3 Change Management

The customer submits a change request form to customer support. Customer support deems whether the change is approved or disapproved. When approved, the request is sent to the

software manager. The software manager then leads the programmers in the development of the feature.

2.1.4 Release Management

All configuration files, data files, and installation instruction files must be updated. A pre-final version of the software will be sent to quality assurance to check that the software is functional and satisfactory. If any problems arise during quality assurance, the software is sent back for changes. If the product passes, then it is released to the Google Play Store.

2.2 Responsibilities

Customer Support:
Purpose: Satisfy customer requests and concerns.
Objective: Receive customer feedback and report feedback to managers.
Membership: Public Relations
Period of Effectivity: Release Phase

Data Programmer:
Purpose: To get data from gameplay and calculate statistics and graphs.
Objective: Create statistics calculation and graphic display of statistics.
Membership: Programmer
Period of Effectivity: Development Phase

Game Programmer:
Purpose: To provide games whose gameplay corresponds to a selected task and send results to the user's account.
Objective: Create game functionality and send game results to the active user's progression data.
Membership: Programmer
Period of Effectivity: Development Phase

Graphic Designer:
Purpose:To make aesthetically pleasing appearance of GUI elements.
Objective: Create art, font, and color schemes for the GUIs of the software.
Membership: Artist
Period of Effectivity: Development Phase

GUI Programmer:
Purpose: To provide a simple to use way of accessing the software's various functions.
Objective: Create intuitive layouts and link GUI elements to their respective operation (e.g. "Play Game" button opens the game GUI and starts the game).

Membership: Programmer
Period of Effectivity: Development Phase

Product Manager:
Purpose: Oversee all development and maintenance of the software.
Objective: Decide on new objectives and track progress of project development.
Membership: Manager:
Period of Effectivity: All Phases

Software Manager:
Purpose: To ensure that the software is being developed as envisioned.
Objective: Track the progress of code development and lead code development.
Membership: Manager
Period of Effectivity: Development Phase

Quality Assurance:
Purpose: To ensure that the software is functional and as envisioned.
Objective: Perform various tests on every aspect of the software and document any bugs, failures, or unexpected operations.
Membership: Software Tester
Period of Effectivity: System Testing Phase

3. Activities

3.1 Configuration Identification

Libraries
MPAndroidChart

3.2 Configuration Control

3.2.1 Requesting Changes

Submit requests via email to the software manager. Format the email subject as “Change Request: [Description]”. Keep the description brief. In the email body, format the request as what you want to change/add and why you want to change/add it.

3.2.2 Evaluating Changes

Changes will be evaluated based on the number of beneficiaries the change would yield.

3.2.3 Approving and Disapproving Changes

Changes will be approved if:

a) It provides significant improvement to the software.
b) It adds a necessary feature.
c) It improves security.
d) It improves privacy.

Changes will be disapproved if:

a) It is redundant or,
b) It removes a significant amount of pre-existing features or,
c) It compromises security, safety, or privacy.

3.2.4 Implementing Changes

Changes will begin on a separate branch. Once complete, it will send a push request to master. It will then be evaluated by the software manager if the change is satisfactory, and merged to master if it passes.

3.3 Configuration Status Accounting

ZenHub is to be used for tracking development progress. Tasks are put in one of six categories: New Issues, Icebox, Backlog, In Progress, Review & QA, and Done.

New Issues: Tasks to do for the next sprint.
Icebox: Low Priority Tasks for the current sprint.
Backlog: High Priority Tasks for the current sprint.
In Progress: Tasks currently being worked on.
Review & QA: Tasks that have finished and are awaiting approval from the manager and QA team.
Done: Tasks completed during this sprint.

GitHub shall be used to manage storage of the project.

3.4 Configuration Evaluations and Review

Managers will meet to discuss new objectives and set schedules for development. Managers will review new or changed configuration items before committing to the project development.

3.5 Interface Control

The software manager is in charge of coordinating the changes of configuration items with changes to interfacing items outside of the scope of the configuration management plan.

3.6 Subcontractor / Vendor Control

Subcontractor and vendor work will be gathered in an external directory by the software manager. The software manager will direct the programmers to incorporate the external work into the software project as appropriate.

3.7 Release Management and Delivery

When the product is ready for release, it will be released with a version number in the format “X.Y.Z”, where X is the external release number (scope is the customer), Y is the internal release number (scope is QA), and Z is the revision number (scope is programmers). A changelog unique to the latest release will be attached and document the new additions followed by changes made.

The product manager will deliver the product to the Google Play Store.

4. Schedule

Date	Milestone
4/10	Develop the GUIs necessary for account creation. Develop statistics display.
4/17	Develop the “Shape” game. Develop User Database.
4/24	Connect game data to user data. Connect user data to statistics display.
TBD	**TBD**

5. Resources

Android Studio is needed to develop the software. Testing will be done using an Android device, which can be either through USB input or through emulation, which Android Studio provides.

GitHub is needed for version control of the software.

6. Plan Maintenance

The project manager is responsible for monitoring the plan. Updates to the plan should happen at the end of the month. Meetings will be held to discuss plan changes. Plans will be evaluated based on the benefits and problems it will cause. If the plan yields more benefits than problems, it shall be approved. The project manager will apply changes to the document and the software manager will communicate these changes with the development team during the next meeting.