**Awesome Alphabet**

**Project Proposal**

Vivek Goyal - Michael Grant - Jaleel Kazi - Mark Musante - Levi Paul

Professor Yuting Zhang

Boston University

Spring 2013

MET-CS-673-W4

Software Engineering

February 5, 2013

**Introduction**

For our project, we will be developing an educational software product known as Awesome Alphabet. Awesome Alphabet will be a desktop-computer-based learning tool designed for 4-6 year old children that assists them in learning the English alphabet. As we all know, learning the alphabet is the first step in developing a child’s reading and writing skills. The goal of this project is to create an easy to use and engaging application that promotes early development of this critical knowledge.

The main features of Awesome Alphabet include showing children all the letters in the alphabet; providing pictures and sounds to associate letters with words and objects; and providing parents with the ability to upload custom pictures and sounds. Time permitting, additional features may include a game to test the child’s knowledge, monitoring of the child’s progress, different themes when presenting letters (i.e. animals, transportation, etc.), multiple user accounts for multiple children, and a password system to prevent children from accessing parental features.

**Related Work**

There are many existing applications that attempt to teach children the alphabet. Some of these include:

1. Teach Me ABCs, funschool.kaboose.com – An activity where a child can click on letters of the alphabet. Each time a letter is clicked, a letter page appears. On each letter page, the letter is displayed in both upper and lower case along with three pictures. Clicking on a letter or item in the picture causes the name of the letter or item to be spoken. For the letter ‘A’, “Aa” is displayed along with pictures of an ant, apple, and axe.
2. Learn Letters, yourchildlearns.com – An activity where a child is presented with all letters of the alphabet in order. Each time a letter is presented, the child hears its name. To proceed to the next letter, the child must fill an outline of the letter with pieces that are scattered around the screen (i.e. for an ‘a’, the child must click and drag a circle and a vertical bar to the letter outline).
3. Alphabet Action, learningplanet.com – An activity where a child may click on letters in the alphabet. When a letter is clicked, its name is pronounced, the letter is shown, and a picture, along with its associated word, is displayed. For example, if the child clicks on ‘a’, “aA” is displayed, along with a picture of an apple and the word “apple”.
4. Learn Your ABCs, fisher-price.com – An activity where the letters of the alphabet are displayed in order and pronounced.
5. Alphabet Whack-A-Mole, playkidsgames.com – A game where moles pop-up out of the ground with a letter. The child must hit the moles in alphabetical order by clicking on them.
6. The Frog Game, playkidsgames.com – A game where letters move across the top of the screen. When the correct letter is above the frog, the child must click on the screen for the frog to eat it. The letters must be eaten in alphabetical order.
7. Kids Alphabet - An Android app that shows the letters of the alphabet. On each letter page, the letter is displayed along with a word and picture associated with that word. The letter is also pronounced.
8. ABC for Kids All Alphabet Free, play.google.com – An Android app that is similar to Kids Alphabet.
9. ABCs, play.google.com – An Android app that shows virtual flash cards of letters in the alphabet. On the front of each card, there is a letter. On the back, there is a picture and word. The application is able to pronounce each letter and provides a way for parents to make their own flash cards with customized pictures and text.

**Project Tools, Process, and Target Systems**

Awesome Alphabet will be developed using the Java programming language. Several members of the group have prior experience with this language and it will be easy to learn for those who have never used it. To assist with the coding process, the Eclipse Integrated Development Environment (IDE) will be used. This IDE was chosen due to its extensive collection of plugins, widespread use in the industry, large community support, and the fact that it is free to download and install.

Initially, there was a desire to make Awesome Alphabet run on an Apple IPad or Android tablet. A tablet would be the most logical hardware interface since it is touch-centric and, therefore, easy to use for a child. Unfortunately, due to limited hardware availability and virtual machine issues that were encountered while researching these options, we have decided to target the desktop Java Runtime Environment (JRE) instead. As a result, Awesome Alphabet will run on any Operating System (OS) that has a supported JRE, including Windows, Linux, and Mac OS X. To implement the user interface (UI), we will use the Java Swing library. One of our design goals will be to separate the UI from the back-end logic to allow us to easily port this program to a tablet OS in the future.

To make the design and development process as smooth as possible, we will be using an agile-like iterative process. With this process, we will add functions to the application incrementally and be able to provide working demos at the end of each iteration.

In addition to the items mentioned above, we will be using the following tools to assist us with the software engineering process:

1. PivotalTracker (<https://www.pivotaltracker.com/>) – A tool for creating user stories, managing workloads, and tracking completion during development iterations. Our public project is located at <https://www.pivotaltracker.com/projects/737427/stories>.
2. Git – A distributed version control system for managing source code.
3. GitHub (<http://github.com>) - A website that provides free public hosting of source code and documents. It supports Git and contains issue tracking / bug tracking tools. Our project repositories are located at <http://github.com/cs673>.
4. EGit – A plugin that will allow us to use Git and GitHub directly from the Eclipse IDE.
5. Google Drive (<https://drive.google.com/a/bu.edu/>) – A website that will host several project documents.
6. Ant or Maven – Build automation tools.
7. Emma – A code coverage tool to assist with the testing process.
8. PMD, FindBugs, and/or CodePro AnalytiX – These tools analyze the Java source code to find bugs, unused items, etc.
9. UML – A model language that describes objects in a software system and their relationships.
10. JavaRefactor – An Eclipse plugin that assists in the refactoring of Java code.
11. JUnit – A unit testing framework for Java.
12. Log4J – A logging library for Java.
13. MyBatis – A data mapping framework for Java that couples objects to database stored procedures and SQL statements.
14. JavaDoc – A utility for generated API documentation based on tagged comments in the source code.
15. Skype – For conference calls / group meetings.

**Functional Requirements**

The following list details various functional requirements of the Awesome Alphabet educational software. The list is divided into three parts: Mandatory Features, Anticipated Features, and Desired Features. Mandatory Features define basic functionality that must be developed into the finished product; Anticipated Features are high-priority features that we would like to implement, but may be dropped due to time constraints; and finally, Desired Features are low-priority features that may be implemented at the end of the project if time permits.

1. Mandatory Features
   1. An alphabet page where the child can select a letter to learn about.
   2. A picture will be shown and a sound will be played whenever a letter is shown. For example, for the letter ‘C’, an image of a cat may appear and the child will hear “Cat begins with a ‘C’”.
   3. Multiple sounds and images for each letter. A child can click a button on the letter page to see and hear additional pictures and sounds for that letter.
2. Anticipated Features
   1. The ability for a parent to upload custom pictures and sounds to associate with letters.
   2. Different themes when presenting letters (i.e. animals, transportation, etc.).
   3. A game to test the child’s knowledge. In this game, the child will see a picture and hear a word and will need to select the letter that the word begins with. The game may use the currently selected theme (if available).
3. Desired Features
   1. A slideshow of the alphabet. Each letter will be shown for 5 seconds.
   2. Monitoring of the child’s progress.
   3. Multiple user accounts for multiple children.
   4. A password system to prevent children from accessing parental features.
   5. The ability for the parent to configure the application’s appearance.

**Nonfunctional Requirements**

In addition the above functional requirements, Awesome Alphabet will have the following nonfunctional requirements:

1. Interface
   1. The application’s window area will have an aspect ratio between 4:3 and 16:9, which is commensurate with the aspect ratios of most tablets.
   2. Fonts will be no smaller than 11 points.
   3. Cursive fonts will not be used.
   4. If a button contains text, it shall also contain a picture or symbol (since the child cannot read words yet). This requirement does not apply to parental administrative options.
   5. A keyboard may be used for some actions.
2. Performance
   1. Any time a screen transition occurs, the new screen must be shown within two seconds.
   2. If a sound automatically plays when a screen appears, it shall be played within two seconds of the screen appearing.
3. Reliability
   1. The application shall not crash 99.999% of the time.
   2. The response time for any user action shall not degrade by 50% or more on average within one hour of starting the application (due to memory management issues or other issues).
4. Constraints
   1. The application will be no larger than 1GB in size. This does not include content that is uploaded after installation.
   2. Memory usage will be no larger than 512MB on average.
   3. Pictures and sounds will be stored using one or more file formats that are common in the public domain (i.e. JPEG, MP3, etc.). File formats will also be restricted to the limitations of the Java programming language, libraries, and/or program database.
   4. Sounds can be no longer than 20 seconds in duration.
5. Security
   1. If a password security option is implemented:
      1. The child must not be able to access any screen that is supposed to be password protected.
      2. The stored password must be encrypted.
6. Portability
   1. The application will be programmed using the Java programming language and target the Java Runtime Environment. As a result, it will work on any OS that has a supported JRE, including Windows, Linux, and Mac OS X.
7. Error Handling
   1. If a critical error occurs, the user will be notified via a message box.
   2. Any errors will be logged to a log file.
8. Additional Notes:
   1. Performance and Reliability requirements do not apply when there is interference by the operating system and/or other applications. For example, it is permissible that a screen transition takes longer than two seconds if another application is using 100% of the CPU.

**Timeline**

Since this is a class project, it must be completed before the end of the semester. Currently, the project due date is May 2, 2013.

The project will be divided into 3 iterations, each lasting 3-4 weeks. At the end of each iteration, a working product will be delivered. The first iteration will be 3 weeks long and will be finished by February 28, 2013. The remaining two iterations will be 4 weeks long and will be finished by March 28, 2013 and May 2, 2013, respectively. Currently, we plan on working on Mandatory features in the first iteration and both Mandatory and Anticipated features in the second iteration. The third iteration will be reserved for any remaining bug fixes, Desired Features, and finishing any Mandatory or Anticipated features that were not implemented in the first two iterations.

**Deliverables**

By the end of this project there will be many deliverables. This deliverables include, but are not limited to:

* Source Code
* Project Proposal (this document)
* Software Quality Assurance Plan (SQAP)
* Software Project Management Plan (SPMP)
* Software Configuration Management Plan (SCMP)
* Software Requirements Specification (SRS)
* Software Design Document (SDD)
* Software Test Document (STD)
* User’s Manual
* Maintenance Plan
* Software Validation and Verification Plan (SVVP)
* Project Presentations

Some of these deliverables will be provided before the end of the project. Documents may be updated as the project progresses.

**Conclusion**

The above sections discuss estimates and initial approaches to many aspects of the project. As the design and development process progresses, these estimates and approaches will be updated and refined. In the end, by using good software design practices, we hope to have on-time and error-free deliveries and hope to make Awesome Alphabet a feature-rich, fun and engaging application.

**References**

*Alphabet Games*. (2012). Retrieved Feb 03, 2012, from PlayKidsGames.com: http://www.playkidsgames.com/alphabetGames.htm

Apache Software Foundation. (2012). *Apache Log4j 1.2*. Retrieved Feb 05, 2013, from http://logging.apache.org/log4j/1.2/

Apache Software Foundation. (2013, Feb 02). *The Apache Ant Project: Welcome*. Retrieved Feb 02, 2013, from http://ant.apache.org/

Apache Software Foundation. (2013). *Welcome to Apache Maven*. Retrieved Feb 02, 2013, from http://maven.apache.org/

Aruhat Technologies Pvt. Ltd. (n.d.). *ABC for Kids All Alphabet Free*. Retrieved Feb 03, 2013, from Google Play: https://play.google.com/store/apps/details?id=com.aruhat.mobileapps.abc&hl=en

Braude, E. J., & Bernstein, M. E. (2011). *Software Engineering: Modern Approaches* (2nd Edition ed.). John Wiley & Sons, Inc.

Disney. (n.d.). *Teach Me ABCs*. Retrieved Feb 03, 2013, from kaboose.com: http://funschool.kaboose.com/preschool/learn-abcs/games/game\_teach\_me\_abcs.html?g=az\_teachb

Eclipse Foundation. (2013). *About the Eclipse Foundation*. Retrieved Feb 2013, 02, from http://www.eclipse.org/org/

ElegantBits. (n.d.). *Kids Alphabet*. Retrieved Feb 03, 2013, from Google Play: https://play.google.com/store/apps/details?id=org.elegantbits.android.kidsworldexplorer.alphabet.paid&hl=en

Fisher-Price / Mattel, Inc. (2013). *Learn Your ABCs*. Retrieved Feb 03, 2013, from http://www.fisher-price.com/us/fun/games/abc/

Five Pumpkins. (n.d.). *ABCs*. Retrieved Feb 02, 2013, from Google Play: https://play.google.com/store/apps/details?id=com.teneke.letters&hl=en

Git Open Source Community. (2013). *Git*. Retrieved Feb 02, 2013, from http://git-scm.com/

GitHub, Inc. (2013). *GitHub Features*. Retrieved Feb 02, 2013, from https://github.com/features/projects

Google, Inc. (2012). *CodePro Analytix User Guide*. Retrieved Feb 02, 2013, from https://developers.google.com/java-dev-tools/codepro/doc/

jEdit. (n.d.). *JavaRefactor*. Retrieved Feb 02, 2013, from http://plugins.jedit.org/plugins/?JavaRefactor

*JUnit: A programmer-oriented testing framework for Java*. (n.d.). Retrieved Feb 02, 2013, from http://www.junit.org/

Multiple Authors. (2013, Jan 22). *Apache Ant*. Retrieved Feb 02, 2013, from http://en.wikipedia.org/wiki/Apache\_Ant

Multiple Authors. (2013, Jan 21). *Apache Maven*. Retrieved Feb 02, 2013, from http://maven.apache.org/

Multiple Authors. (2013). *EGit*. Retrieved Feb 02, 2013, from http://www.eclipse.org/egit/

Multiple Authors. (2013, Jan 28). *Javadoc*. Retrieved Feb 02, 2013, from http://en.wikipedia.org/wiki/Javadoc

Multiple Authors. (2013, Feb 05). *MyBatis*. Retrieved Feb 05, 2013, from http://en.wikipedia.org/wiki/MyBatis

Multiple Authors. (2013, Feb 01). *Unified Modeling Language*. Retrieved Feb 02, 2013, from http://en.wikipedia.org/wiki/Unified\_Modeling\_Language

Multiple Authors. (n.d.). *MyBatis*. Retrieved Feb 05, 2013, from http://code.google.com/p/mybatis/

Oracle. (n.d.). *Javadoc Tool Home Page*. Retrieved Feb 02, 2013, from http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html

Owl & Mouse Educational Software. (2011). *Learn Letters Online*. Retrieved Feb 03, 2013, from http://www.yourchildlearns.com

Pivotal Labs, Inc. (2013). *PivotalTracker Features*. Retrieved Feb 02, 2013, from http://www.pivotaltracker.com/features

Planet Interactive, Inc. (2012). *Alphabet Action*. Retrieved Feb 02, 2013, from learningplanet.com: http://www.learningplanet.com/act/fl/aact/index.asp

PMD. (2012). *Welcome to PMD*. Retrieved Feb 02, 2013, from http://pmd.sourceforge.net/pmd-5.0.1/

Roubtsov, V. (2006, Jan 14). *EMMA: Frequently Asked Questions*. Retrieved Feb 2013, 02, from http://emma.sourceforge.net/faq.html#faq-N1000B

University of Maryland. (2012, December 10). *FindBugs: Fact Sheet*. Retrieved Feb 02, 2013, from http://findbugs.sourceforge.net/factSheet.html