Hello Future Developers of Trivio!

This document is an overall summary of Trivio to give you guidelines on how to reproduce and continue what we started. Trivio, also known as Trivia Throw Toy, is a ball-shaped toy that vocalizes a random fun fact from the internet when thrown or shaken. The fun facts are generated by scraping websites and are stored in our database hosted on Azure. The toy connects to this database and downloads facts onto its local storage. This ensures an almost instantaneous fact retrieval when a throw or shake is detected.

The toy has three main aspects - software, hardware, and structure. The software includes a server hosted on Azure and an Arduino that gives the toy its functionality. The hardware includes the PCB and additional modules and components. The structure includes the shell and soft outer layer that houses the PCB and all electronic components of the toy. The tools used to accomplish the software, hardware, and structure are the Arduino IDE, Altium Designer, and Solidworks respectively.

The hardware modules used in developing the toy are:

* WiFi Module - Internet connection for server and local connection to the mobile app
* Accelerometer - To detect a throw or shake
* Text-to-Speech - Convert the facts from string to speech
* SD Card Reader - To store and read facts locally on the toy
* Arduino Mega - To combine the functionality of all the modules
* Voltage Booster - To step up the battery voltage from 4V to 9V
* Charging Module - To charge the Lipo battery

Other components used are:

* RGB LED - To notify user of different statuses
* Buttons - For increasing and decreasing the volume as well as setting up the WiFi
* Switch - To turn the toy on and off
* Battery - Rechargeable Samsung battery to power the toy
* Speaker - To hear the facts when vocalised
* Eva foam- To protect the toy

Since the toy will be thrown around a lot, it is essential to make all the electrical components as compact as possible. This reduces possibilities of a loose or broken connection occurring. Therefore, we created a PCB to hold all the modules together and create the connections between them without requiring external wires. The shell of the toy has a plate on which the PCB is held into place. The housing was 3D printed and is coated with a layer of soft eva foam to reduce the shock on the ball if accidentally dropped.

An important thing to look out for is the PCB. The voltage booster failed to work on the PCB, so we had to do those connections externally - more wires. Try to modify the PCB to ensure that the voltage booster will work with it. After going through it several times, we could not figure out what went wrong during the fabrication. The schematics provided are a 100% correct, and the problem is most likely in the PCB1.PcbDoc file.

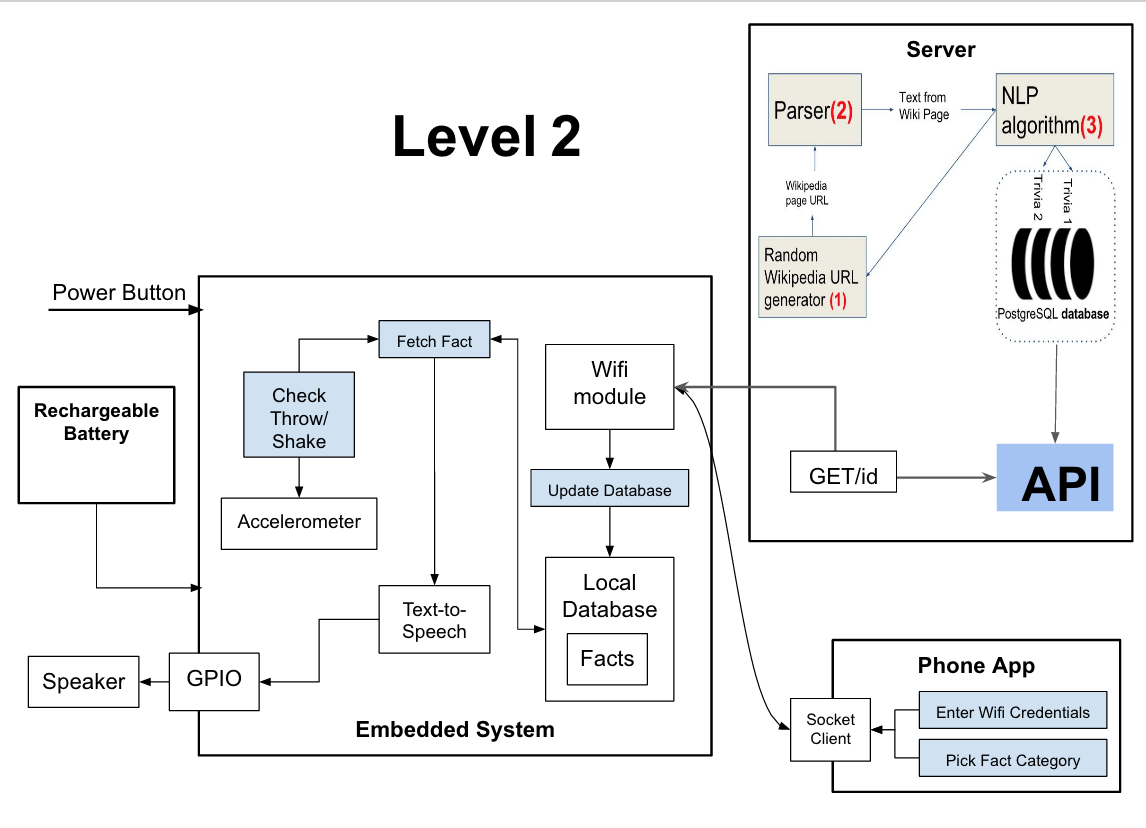
Hardware:

* Only use an Arduino Mega! The Arduino Uno cannot handle Trivio’s program.
* Make sure all the modules work independent of each other.
* When putting all the modules together, ensure that they are connected to the correct pins on the Arduino Mega. Follow the schematics provided in Sheet3.SchDoc and Sheet4.SchDoc.
* When adding the modules to the PCB, ensure that they are soldered on properly. Loose connections can cause glitches and bugs.

There is definitely room for improvement. The PCB can be bettered by recreating the circuitry of all the modules on the PCB instead of just mounting them. This requires expertise in PCB designing as well as detailed datasheets of all the modules and chips used. However, doing this can immensely reduce the size of the PCB, thus reducing the size of the ball.

Software:

The Level 2 block diagram below is essential to understanding the logical flow of the program from detecting a shake to downloading and narrating a new fact. A detailed explanation of these blocks can be found in our testing reports, however this diagram will give you a quick overview of the software.



On the server side, the most important component is the API. Before working on the project, you should use POSTMAN to test the API. Detailed implementation can be found at this link: <http://triviotoy.azurewebsites.net/Help>.

There is however definitely room for improvement in the server code. You will realize that all API endpoints are unauthenticated and passwords are not hashed when stored in the database. You must work on improving server side security. Additionally, currently every user will iterate through the same facts in the same order, however you should consider randomising this.

Our parsing algorithm accepts a web url and the html tag which contains facts as an input parameter. It then returns a list of facts. Since you already have the platform ready, consider implementing a NLP algorithm to extract trivia from a website without knowing its enclosing html tag.

On the database side, we used Entity Framework Code first migrations. Any attempt to change the scheme of the database must only be done using migrations, else the code will fail(If you use sql to modify the schema). If you want to seed the data with default values, enter it in Migrations.Configuration.cs.

Have fun with Trivio!