C E R E B R A
a project of Simple Labs

SCOPE
The aim of this project is visualizing the fMRI data on a 3D graph to increase the understandability of the complex data. The fMRI data includes the brain response of a human in response to some particular circumstances as showing the picture of a red apple.

The main role of the final 3D graph will be visualizing voxels and edges which show the related voxels. Besides graph may include a brain image as background and five main lobes of the brain, which are frontal, parietal, occipital, limbic, temporal lobes, in different colors.

Since the fMRI data is very large and complex, time and space will be main constraints.

The target audience of this project is mostly academicians and medical institutes. Cognitive state representation and visualization of human brain is fundamentally important in neuroanatomy, neurodevelopment, cognitive neuroscience and neuropsychology.

This project will be implemented in Unity 3D Game Engine with using its DOTS libraries. C# will be used as the programming language.

IMPLEMENTATION
Extenisibility is a must for the project. Since this project will be used in a highly active research area, it is essential that novel ideas be implemented easily. Project team aims to achieve this with a highly algorithm and data structure approach.

FINAL PRODUCT
Here are some screenshots of our program.

PACKET CLASS
This class is used as an immediate data format between two Processors. Packet class encapulates all data needed by Processors: voxel coordinates, edge values, etc... It also offers a way to pass named extra data between Processors.

PROCESSOR INTERFACE
This interface defines outlines of Processors and how Processes should be implemented.

PIPELINE CLASS
This class is responsible for chaining Processor operations. A pipeline is an object that the user can save to or load from a file. First, it also enables the user to create his/her own projects. When a Processor is added to the Pipeline, Pipeline object checks whether it is the first Processor to be added, and if it is, it is an input type Processor.

PROCESSOR MANAGER CLASS
This class is responsible for managing Processor selection and generation. This is a static class and its members are all static. Each Processor must register itself with the ProcessorManager. C++ doesn't allow static constructors, which would be used when registering. This is a problem the team is working on.

ACKNOWLEDGEMENTS
We thank to Prof. Fatih Yarman Tunay Vural and her colleagues at the Image Processing Laboratory for their help throughout the project.