The Hackademia Application for our senior design project can be broken into a 3-Tier Architecture consisting of a client, server, and database. At the most basic premise, shown in design diagram D0, the application takes in the interacting users information and the C code snippet to convert. Then, abstracted from the user within the application, Hackademia returns the compiled assembly code and corresponding stack and register animation back to the user.

Removing the abstraction and looking into the application, as shown in design diagram D1, the three-tiered architecture becomes apparent. The top user-interface level, known as the client, is the application layer in which the user interacts with. This is written in React, a common JavaScript library used to write component style interfaces. The second level, known as the server, communicates with the client via request and response calls. When the user inserts the C code to be converted, the client sends a request to the server to fetch the assembly code and data. The server then completes its own logic processing using Node JS, a JavaScript runtime environment. The third level, known as the database, stores data from the server. When the server finishes its logic processing and converts the C code to assembly, it then stores the data within the database. This is written in SQLite, a commonly used relational database management system. Finally, the server sends a response back to the client, which then displays the data for the user.

To further break down the details of the application, as shown in design diagram D2, the specific communication and processing mannerisms are defined. The client and server communicate through HTTP request and response calls, which are allotted by both React and Node JS through the simplification library Express JS. When the user inserts C code and the client sends a request to the server, the server runs its processes deriving and parsing the assembly code as well as the return stack and register values using Python scripting. Then, the server stores this data within the database using a SQL INSERT command. If the user is requesting data that has already been inserted into the database, the server would not run its processes and would instead immediately run a SQL SELECT query to return the data. The database then returns the queried data to the server, and the server sends an HTTP response back to the client.