CHAPTER 3

RESEARCH METHODOLOGY

3.1. Literature Study

This research activity begins with the collection of various literary journals and research related to the reliability test activities of the use of gRPC and RESTAPI. Not only that, various other research journals related to the use of the Go programming language and the gRPC and REST API reliability testing methods are also included to support information and knowledge relevant to research activities. All the journals that have been collected are then used as a reference or reference in carrying out research activities that will be carried out at this time.

3.2. Building gRPC Client and Server Program Code Using the Go

The next stage of the research activity is to develop program code using the gRPC framework which is one of the RPC frameworks using the RPC protocol for the transport process, and prototype for the interface. The development of the program code is carried out using the Go programming language for both the client-side and the server-side. The gRPC framework is used in research activities because this is in line with the objectives of the research activity itself, namely the reliability test between gRPC and the REST API.

From the client-side, the gRPC program code is built using the import package google.golang.org/grpc and several other packages, wherein the code section of this program several port variables are declared that will be used for communication to the server.

From the server-side, the program code is built using the import package google.golang.org/grpc and several other packages, wherein the server program code the port variable is declared for the serial communication process from the client to the server which will then be forwarded to the database server.

3.3 Building Client Program Code and Server REST API Using the Go

After finishing creating the program code from the client and server-side of gRPC, then the program code development from the client and server-side of the REST API is carried out. The REST API does not import google.golang.org/grpc, but the program code can be directly built on the server itself by using the "net/http" package and several other packages related to the program code, and some variable declarations and structs for the communication process.

From the client-side, the program code is built starting with the process of importing the "net/http" package and several other packages related to the serial communication process to the server, as well as the struct declaration that will be sent to the server.

From the server-side, the program code is built starting with the process of importing the "net/http" package and several other packages related to the serial communication process from the client to the server, and several other program commands that are used to receive variable values sent by the client to the server. to be forwarded to the database server.

3.4. Build Local Server

The next stage of the research activity is to build a local server using a Golang Web Server that can be accessed by gRPC or REST API. Unlike other programming languages, where the server is separate and requires additional installation processes, in the Go programming language, the server or web server is in the Go programming language itself, so that the server can be accessed directly by programs or applications developed using the Go programming language without the need to do the additional installation process.

3.5. Build Database Using PostgreSQL

After the local server has been built, the next step is to create a database using a PostgreSQL database. Because the "database/sql" package on the Golang Web Server is a generic interface in the Go programming language, connecting to the PostgreSQL database is done by importing drive pq ("github.com/lib/pq"), and renaming it, when a new connection is established by the server.

3.6. Testing

The two program codes that have been created (gRPC and REST API) are then tested using JMeter with several observation variables that have been determined by the researcher related to the value of data transfer performance between gRPC and REST API.

3.7. Analysis

The test result is the value of the data transfer performance value sent from gRPC and the RES API to the Server whose results are obtained from the measurement results using the JMeter measuring device. The results are then analyzed and displayed in a comparison of the values in tabulation and graphically, to provide information about the data transfer performance and efficiency values of gRPC and RESTAPI. The data from the test results that have been displayed using tabulations and graphics are then explained qualitatively, both in terms of advantages and disadvantages, improvements can be made in further research.