# HTTP2 Server

## Components:

1. **Connection Handler**

* Manages TCP connections from clients.
* Validates HTTP/2 connection preface.
* Initializes client-specific settings.

1. **Frame Processor**

* Parses incoming HTTP/2 frames (HEADERS, DATA, SETTINGS,….).
* Validates frame types, flags, and lengths.
* Delegates frame processing to the appropriate handler.

A close-up of a computer code

Description automatically generated

1. **Stream Manager**

* Manages the lifecycle of streams (creation, activity, closure).
* Handles stream prioritization and dependencies.
* Ensures compliance with HTTP/2 multiplexing rules.

A diagram of a flowchart

Description automatically generated

1. **HPACK Compression/Decompression Module**

* Implements header compression and decompression using static and dynamic tables.
* Encodes and decodes HTTP header blocks as specified in RFC 7541.

A close-up of a computer code

Description automatically generated

1. **Flow Control Module**

* Manages connection-level and stream-level flow control.
* Processes WINDOW\_UPDATE frames and adjusts flow control windows accordingly.

1. **Error Handler**

* Detects and handles protocol errors such as malformed frames or invalid stream states.
* Sends appropriate HTTP/2 error codes (RST\_STREAM, GOAWAY, ……).

## Interaction Between Components:

1. **Connection Handler**

* **Input**: Incoming client TCP connection.
* **Tasks**:
  + Read and validate the HTTP/2 connection preface.
  + Exchange initial SETTINGS frames with the client.
  + Forward incoming frames to the Frame Processor.
* **Output**: Validated and parsed frames passed to the Frame Processor.

1. **Frame Processor**

* **Input**: Raw frames received from the Connection Handler.
* **Tasks**:
  + Parse frame headers and payloads.
  + Dispatch frames to the appropriate modules
  + Enforce protocol rules, such as stream ID validation.
* **Output**: Processed frames forwarded to respective modules

1. **Stream Manager**

* **Input**: HEADERS and DATA frames.
* **Tasks**:
  + Manage stream states (idle, open, closed).
  + Handle prioritization and dependency hierarchies.
  + Forward stream-specific data to the application layer.
* **Output**: Encoded/decoded HTTP/2 responses sent to the Frame Processor.

1. **HPACK Compression/Decompression Module**

* **Input**: Header blocks from HEADERS frames.
* **Tasks**:
  + Compress HTTP/2 response headers into header blocks.
  + Decompress client header blocks and populate dynamic tables.
* **Output**: Compressed or decompressed header data passed to the Stream Manager.

1. **Flow Control Module**

* **Input**: WINDOW\_UPDATE frames.
* **Tasks**:
  + Manage flow control windows for streams and the connection.
  + Ensure that outgoing DATA frames respect flow control limits.
* **Output**: Adjusted flow control windows passed to the Frame Processor.

1. **Error Handler**

* **Input**: Detected protocol violations or internal errors.
* **Tasks**:
  + Log errors and send appropriate HTTP/2 error frames (e.g., GOAWAY).
  + Gracefully terminate streams or the connection.

## System Communication Protocols:

**1. HTTP/2 Frames**

* The server will implement the frame structure defined in RFC 7540, with each frame comprising:
  + **9-byte header**: Includes length, type, flags, and stream ID.
  + **Variable-length payload**: Depends on the frame type.

**2. Header Compression**

* Use HPACK (RFC 7541) to compress and decompress HTTP/2 headers.
* Maintain static and dynamic tables for efficient header encoding/decoding.

**3. Multiplexing and Flow Control**

* Use the stream identifier to manage multiple streams over the same connection.
* Respect flow control windows to ensure efficient data transmission.

## Key Functionalities of Web Server:

1. **Handle Client Connections**

* Accept incoming connections from clients
* Maintain persistent connections (keep-alive) to avoid re-establishing connections for each request.

1. **Protocol Handling (HTTP/2)**

* Process HTTP/2 frames
* Compress HTTP headers using HPACK to reduce bandwidth usage.

1. **Error Handling and Custom Error Pages**

* Return proper HTTP status codes.
* Provide custom error pages.