Project planning and processes:

| | |
|---|---|
| Install Rust & Set Up Project | creates a new project named my_web_app<br><br>cargo new my_web_app<br>cd my_web_app |
| Add Axum ( Web Server) | `axum` helps **handle web requests**.<br>`tokio` lets you **run async code** (important for web servers).<br>cargo add axum --features "full"<br>cargo add tokio --features "full"<br>cargo add serde_json # For JSON responses<br>cargo checkcargo update |
| Write Web Server Code in src/main.rs | Creates a web server<br><br>Defines two routes:<br>/ → Shows "Welcome to my Rust Web App!".<br>/hello → Returns JSON { "message": "Hello, world!" }.<br>Runs the server on localhost:3000. |
| | |
| Run the Web App | cargo run<br><br>Open browser and visit:<br><br>http://127.0.0.1:3000/ → Shows welcome message.<br>http://127.0.0.1:3000/hello → Returns JSON. |
| Save Work in Git & Push to GitHub | git init  # Start tracking changes<br>git add .  # Add all files<br>git commit -m "Initial commit"<br>git branch -M main  # Rename branch to main<br>git remote add origin <your-github-repo-url>  # Link to GitHub<br>git push -u origin main  # Push your code online |
| | |
| Imports & Dependency | `use axum::{Router, routing::get, Json}; // Axum handles routing & web requests`<br>`use std::net::SocketAddr; // For defining server address`<br>`use tokio; // Asynchronous runtime for handling async` |

| | |
|---|---|
| | ```tasks<br>use tokio::net::TcpListener; // For creating a TCP listener```<br>**axum::Router**: Defines routes (URLs) for our API.<br>**axum::routing::get**: Specifies HTTP GET routes.<br>**axum::Json**: Used to return JSON responses.<br>**std::net::SocketAddr**: Represents the server's IP address and port.<br>**tokio**: Rust's asynchronous runtime.<br>**tokio::net::TcpListener**: Creates a TCP listener to accept connections. |
| Main Function | ```#[tokio::main] // Marks this function as async so Rust can run it properly<br>async fn main() {```<br>**#[tokio::main]**: This tells Rust to use Tokio's async runtime.<br>**async fn main()**: The main function is asynchronous. |
| Creating the Router | ```let app = Router::new()<br>        .route("/", get(root)) // Route for the home page "/"<br>        .route("/hello", get(hello)); // Route for "/hello" which returns JSON```<br>**Router::new()**: Creates a new router.<br><br>**.route("/", get(root))**: Handles GET requests to / with the root function.<br><br>**.route("/hello", get(hello))**: Handles GET requests to /hello with the hello function. |
| Defining the Server Address | ```// Define server address (localhost:3000)<br>    let addr = SocketAddr::from(([127, 0, 0, 1], 3000));<br>    println!("Server running on http://{}", addr); // Print server URL to console```<br>**SocketAddr::from(([127, 0, 0, 1], 3000))**: Creates an IP address 127.0.0.1:3000 (localhost).<br>**println!**: Prints the server URL to the console. |

| | |
|---|---|
| Starting the Server | ```rust<br>// ✅ Alternative way to start the server using Axum<br>let listener = TcpListener::bind(addr).await.unwrap();<br>axum::serve(listener, app).await.unwrap();<br>}<br>```<br>**`TcpListener::bind(addr).await.unwrap();`**: Binds the TCP server to the specified address.<br>**`axum::serve(listener, app).await.unwrap();`**: Starts the Axum server and waits for requests. |
| Handling the Root Route (/)<br>rust<br>Copy<br>Edit | ```rust<br>// Function that handles requests to "/"<br>async fn root() -> &'static str {<br>"Welcome to my Rust Web App!" // Returns a simple text response<br>```<br>**Returns a plain text response** `"Welcome to my Rust Web App!"` to users visiting `/`. |
| Handling the `/hello` Route | ```rust<br>// Function that handles requests to "/hello"<br>async fn hello() -> Json<serde_json::Value> {<br>// Create a JSON response<br>let data = serde_json::json!({ "message": "Hello, world!" });<br>Json(data) // Return JSON response<br>}<br>```<br>**`serde_json::json!`**: Creates a JSON object `{ "message": "Hello, world!" }`.<br>**Returns a JSON response** when users visit `/hello`. |
| Cargo.toml (Dependencies & Configuration) | ```toml<br>[package]<br>name = "my_web_app"<br>version = "0.1.0"<br>edition = "2021"<br><br>[dependencies]<br>axum = { version = "0.8.1", features = ["json"] }<br>hyper = { version = "0.14.32", features = ["server"] } # Enable the `server` feature<br>serde_json = "1.0.138"<br>tokio = { version = "1.43.0", features = ["full"] }<br>```<br>**`axum`**: The web framework.<br>**`hyper`**: Used internally by Axum for HTTP handling.<br>**`serde_json`**: For working with JSON data. |

| | |
|---|---|
| | `tokio`: Asynchronous runtime (required for Axum). |
| | |
| | |