

**File: server/index.ts****Type: ts | Size: 1921 bytes**

```
import express, { type Request, Response, NextFunction } from "express";
import { registerRoutes } from "./routes";
import { setupVite, serveStatic, log } from "./vite";
const app = express();
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use((req, res, next) => {
  const start = Date.now();
  const path = req.path;
  let capturedJsonResponse: Record | undefined = undefined;
  const originalResJson = res.json;
  res.json = function (bodyJson, ...args) {
    capturedJsonResponse = bodyJson;
    return originalResJson.apply(res, [bodyJson, ...args]);
  };
  res.on("finish", () => {
    const duration = Date.now() - start;
    if (path.startsWith("/api")) {
      let logLine = `${req.method} ${path} ${res.statusCode} in ${duration}ms`;
      if (capturedJsonResponse) {
        logLine += ` :: ${JSON.stringify(capturedJsonResponse)}`;
      }
      if (logLine.length > 80) {
        logLine = logLine.slice(0, 79) + "...";
      }
      log(logLine);
    }
  });
  next();
});
(async () => {
  const server = await registerRoutes(app);
  app.use((err: any, _req: Request, res: Response, _next: NextFunction) => {
    const status = err.status || err.statusCode || 500;
    const message = err.message || "Internal Server Error";
    res.status(status).json({ message });
    throw err;
  });
  // importantly only setup vite in development and after
  // setting up all the other routes so the catch-all route
  // doesn't interfere with the other routes
  if (app.get("env") === "development") {
    await setupVite(app, server);
  } else {
    serveStatic(app);
  }
  // ALWAYS serve the app on port 5000
  // this serves both the API and the client.
  // It is the only port that is not firewalled.
  const port = 5000;
  server.listen({
    port,
    host: "0.0.0.0",
    reusePort: true,
  }, () => {
    log(`serving on port ${port}`);
  });
})();
```