توضیح کد.
در ابتدا فایل کافیگ خوانده میشود و متغییرها مقدار دهی میشود. سپس کارهای مرتبط با socket انجام میشود. (چه در windows چه در unix) سپس هر کانکشنی که برقرار میشود توسط تابع accept دریافت شده و مقادیرش در یک آرایه گلوبال قرار میگیرد و شماره‌ی آن خانه به تابع handle_request داده میشود.

تابع هندل ریکوئست آن درخواست را جواب داده و همچنین لاگ خود را در خانه‌ی متناظر قرار داده و اجرایش متوقف میشود. با این حال در هر کدام از حالات مقداری تفاوت وجود دارد که در ادامه برسی میکنیم.

حالت multi thread:
در این حالت بعد از accept یک ترد ساخته شده که مسئول اجرای تابع handle_request است. همچنین برای محدود کردن تعداد thread ها. یک آرایه داریم که میگویید هر ترد در حال اجرا است یا خیر، و با آمدن یک ریکوئست جدید اولین ترد خالی را به آن اختصاص میدهیم. همچنین قابل از شروع اجرای یک ترد چک میکنیم اگر اجرا شده بود، لاگ آن را ذخیره کنیم و سپس اجرای ترد جدید را شروع کنیم.

حالت multi proc:
در این جا هم مثل قبل عمل میکنیم با یک تفاوت، در این جا باید socket ها در هر دو پروسس فرزند و والد بسته شود تا سایت لود کردن را متوقف کند. برای همین بعد از fork کردن، سوکت را طرف والد در همان زمان میبندیم تا مشکلی ایجاد نکند. در تابع فرزند هم تابع handle_request را صدا میزنیم و بعد از اتمام کار آن خروجی این تابع را در یک pipe مینوسیم. به طور مشبه در پردازه‌ی پدر هنگامی که این پروسس قرار بود فعال شود، ابتدا لاگ‌های سری پیش اجرا را خالی میکند. در این حالت با پر شدن pipe متوجه میشویم که کار اجرای پردازه تمام شده است.

در زمان اجرا بر روی windows هم از لایبری `winsock2.h` استفاده میکنیم که لایبری ویندوز برای کارهای شبکه است. (در واقع لایبری استاندارد ویندوز برای باز کردن سوکت است و لایبری اضافه حساب نمیشود.