সফটওয়্যার ইঞ্জিনিয়ারিং আলাপ

আহমেদ শামীম হাসান

অদম্য প্রকাশ

উৎসর্গ

আমার আব্বা-আম্মাকে।
রব্বির হাম হুমা কামা রব্বা ইয়ানি সগিরা।

Hasin Hayder
Co-founder & Head of Ideas
HappyMonster

অভিমত

অনেকের কাছে মনে হতে পারে দুই লাইন কোড লিখতে পারলেই সফটওয়্যার ইঞ্জিনিয়ার হওয়া যায়। আবার কারও কারও কাছে সঠিক গাইডেন্সের অভাবে সফটওয়্যার ইঞ্জিনিয়ারিং জিনিসটা আসলে কী সেটা স্পষ্ট হয় না কখনো। অনেকের মনে নানা রকম প্রশ্ন আসে, যেগুলোর সঠিক উত্তর মেলে না অনেক সময়েই। এই দিক দিয়ে শামীমের লেখা এই বইটা এসব দরকারি নানা রকম প্রশ্নোত্তরে ঠাসা চমৎকার একটা বই। অনেকটা 'পড়তে বসলে শেষ না করে ওঠা যায় না টাইপের' একটা বই। সফটওয়্যার ইঞ্জিনিয়ারিং যে বিশাল একটা জিনিস, এর আগে-পিছে আরও অনেক কিছু জিনিস আছে জানার - শামীম সেগুলোই তুলে এনেছে দারুণ এই বইয়ে। এই বইয়ে আপনি মোটেই শিখবেন না কী করলে হাজার হাজার ডলার উপার্জন করা যায়। বরং আপনি শিখবেন কী করলে এমন একটা অবস্থানে পৌঁছানো সম্ভব যেখান থেকে একটা ডিসেন্ট ইনকাম না করতে পারাটাই বরং আশ্চর্যের বিষয়! অনেকগুলো দারুণ দারুণ "কী" এবং "কেন" প্রশ্নের উত্তর নিয়ে এই বইয়ে শামীমের প্রাঞ্জল এবং সাবলীল উপস্থাপনা নতুনদের জন্য তো বটেই, পুরাতনদেরও অনেক সাহায্য করবে প্রোগ্রামিং, সফটওয়্যার ডেভেলপমেন্ট এবং সফটওয়্যার ইঞ্জিনিয়ারিং নিয়ে নিজেদের অমূলক বা ভ্রান্ত ধারণাগুলো ঝেড়ে ফেলে একদম সময়োপযোগী হয়ে উঠতে।

Ziaul Haque Zia
Vice President of Software Engineering
bKash Limited

অভিমত

আমরা যারা সফটওয়্যার ইঞ্জিনিয়ারিং লাইনে ক্যারিয়ার শুরু করতে যাচ্ছি বা সফটওয়্যার ইঞ্জিনিয়ারিংয়ে অলরেডি ক্যারিয়ার শুরু করেছি তাদের প্রায় সবারই মূল কনসার্ন থাকে প্রোগ্রামিং বা কোডিং নিয়ে। আমরা বিভিন্ন প্রোগ্রামিং বই, টিউটোরিয়াল, কোর্স ফলো করি বা সেগুলো থেকেই টেকনিক্যাল জিনিসগুলো শিখি। কিন্তু কেবল প্রোগ্রামিং ল্যাঙ্গুয়েজ বা কোডিং শিখেই ভালো সফটওয়্যার ইঞ্জিনিয়ার হওয়া যায় না। এর সঙ্গে আরও কিছু ফান্ডামেন্টাল কনসেপ্ট বা নিজের চিন্তাভাবনা/মাইন্ডসেটেরও কিছু পরিবর্তন দরকার পড়ে। তা না হলে আমাদের এই সফটওয়্যার ইঞ্জিনিয়ারিং ক্যারিয়ার খুব বেশি লম্বা হবে না, বা এই লাইনে খুব বেশি ভালো কিছু করাও সম্ভব হবে না। এই বইতে শামীম এমন কিছু বিষয়কে একত্র করেছে যা কিনা সফটওয়্যার ইঞ্জিনিয়ারিংয়ে ক্যারিয়ার শুরু করেছে বা করবে এমন সবারই জানা থাকা দরকার। এই বইতে প্রোগ্রামিং ল্যাঙ্গুয়েজ, সফটওয়্যার ইঞ্জিনিয়ারিং এর টেকনিক্যাল কোনো টপিক শেখানো হয়নি। বরং এই বইতে এমন বিষয়গুলোই আলোচনা করা হয়েছে যেগুলো নিয়ে আমরা সফটওয়্যার ইঞ্জিনিয়াররা অনেক ক্ষেত্রেই ডিফিকাল্টি ফেইস করি বা চিন্তা করি। আবার এমন অনেক কিছুও এই বইতে এসেছে যেগুলো নিয়ে আমরা চিন্তা করি না অথচ চিন্তা করা উচিত। এই বইতে কোনো নির্দিষ্ট প্রোগ্রামিং ল্যাঙ্গুয়েজ শেখা কিংবা কীভাবে ফ্রিল্যান্সিং করে টাকা কামাবেন- এ রকম চটকদার কথাবার্তা বলা হয়নি। বরং শামীম তার ব্যক্তিগত অভিজ্ঞতা, শিক্ষা এবং অন্যদের থেকে অর্জিত জ্ঞানের আলোকে একজন সফল সফটওয়্যার ইঞ্জিনিয়ার হওয়ার জন্য যেই গুণ এবং যোগ্যতাগুলো অর্জন করা দরকার সেগুলো নিয়েই এখানে আলোচনা করেছে। আমার নিজের দীর্ঘ অভিজ্ঞতার আলোকে আমি এই বইয়ের লেখাগুলোর বাস্তবতা এবং প্রয়োজনীয়তা খুব ভালোভাবেই উপলব্ধি করতে পারি। আমি ব্যক্তিগতভাবে মনে করি যারা সফটওয়্যার ইঞ্জিনিয়ারিংয়ে ভালো ক্যারিয়ার গড়তে চায় তাদের সবারই এই ব্যাপারগুলো অবশ্যই জানা জরুরি। বইয়ের সব পাঠককে অগ্রিম শুভেচ্ছা।

লেখকের কথা

সকল প্রশংসা মহান রাব্বুল আ'লামিনের যিনি আমাদের সৃষ্টি করেছেন এবং জ্ঞান দান করে সম্মানিত করেছেন সব সৃষ্টিকুলের মধ্যে।

সফটওয়্যার ইঞ্জিনিয়ারিংয়ে ক্যারিয়ার শুরুর পর থেকে নিয়ে আজ পর্যন্ত দেশি, বিদেশি বিভিন্ন কোম্পানিতে কাজের সুযোগ হয়েছে আমার, আলহামদুলিল্লাহ। বিভিন্ন রকম মানুষের সঙ্গে, টিমের সঙ্গে কাজ করতে গিয়ে অনেক কিছু শিখেছি। এ ছাড়াও ইঞ্জিনিয়ারিং রিলেটেড বিভিন্ন বইপত্র, আর্টিকেলও পড়া হয় নিয়মিত। এসব কিছুর ওপর ভিত্তি করে আমি যখন নিজেকে পর্যালোচনা করি, তখন বুঝি যে এমন কিছু বিষয় আছে যেগুলো ক্যারিয়ারের শুরুর দিকে জানা থাকলে বা শিখলে আমি অনেক রকম ভুল এড়াতে পারতাম। আরও ভালোভাবে কাজ করতে পারতাম। বিভিন্ন সময়ে কমিউনিটির জুনিয়র, সিনিয়রদের সঙ্গে আলাপ-আলোচনা থেকেও দেখেছি আমরা অনেকেই এই জিনিসগুলো না জানার কারণে বিভিন্ন সময়ে বিভিন্ন রকম সমস্যার মুখোমুখি হই।

আমরা যারা টেক ইন্ডাস্ট্রিতে কাজ করছি, আমাদের মধ্যে শুরু থেকেই একটা ধারণা তৈরি হয়ে যায় যে কেবল 'টেকনিক্যাল' বিষয়গুলো ভালোভাবে জানলেই হয়তো ক্যারিয়ারে ভালো করা সম্ভব। কিন্তু সময়ের সঙ্গে সঙ্গে আমরা বুঝতে পারি আমাদের এই ধারণা কতটা ভুল। কেউ কেউ হয়তো একটু আগে আগে বুঝতে পারে এবং নিজেকে সামলে নিতে পারে। কারও কারও হয়তো একটু দেরি হয়ে যায়। আবার কেউ কেউ হয়তো বুঝতেই পারেন না কেন তিনি টেকনিক্যালি এত ভালো হওয়ার পরেও ক্যারিয়ারে ভালো কিছু করতে পারছেন না।

শুধু সফটওয়্যার ইঞ্জিনিয়ারিং না বরং প্রতিটা সেক্টরেই ভালো করার জন্য এমন কিছু ব্যাপার আছে, যেগুলো আমাদের একাডেমিকভাবে শেখানো হয় না। আবার এগুলোর গুরুত্ব নিয়ে আমাদের কনসার্ন না থাকায় আমরাও বুঝি না এগুলো যে শেখা উচিত। অথচ আপনি ইন্ডাস্ট্রি লিডার বা সিনিয়রদের সঙ্গে কথা বলে যদি জিজ্ঞেস করেন তাদের সাফল্যের পেছনে কি কেবল তাদের টেকনিক্যাল স্কিলই যথেষ্ট ছিল কি না, তাহলে আপনি দেখবেন যে তারা এমন কিছু বিষয়ের উল্লেখ করবেন যেটা হয়তো আপনি চিন্তাই করেননি কখনো।

ইঞ্জিনিয়ারিং মাইন্ডসেট, সফট স্কিলস, লার্নিং টেকনিকসহ ইত্যাদি বিষয়ে ইংরেজিতে বিভিন্ন রকম কন্টেন্ট থাকলেও বাংলায় তেমন কোনো ভালো লেখা বা আলোচনা চোখে পড়েনি। এ কারণেই এই বিষয়গুলো নিয়ে ছোট ছোট টপিক আকারে ফেসবুকে লেখা শুরু করেছিলাম। আলহামদুলিল্লাহ, সবাই বেশ ভালোভাবে লেখাগুলো গ্রহণ করেন এবং লেখাগুলোর সংকলিত রূপ বই আকারে নিয়ে আসার অনুরোধ করেন। তাদের উৎসাহেই এই লেখাগুলোকে আরেকটু সংশোধন, পরিমার্জন করে আজকের এই বইয়ের রূপ দেওয়া।

শ্রদ্ধেয় হাসিন হায়দার ভাই এবং জিয়াউল হক জিয়া ভাইকে আন্তরিক ধন্যবাদ বইয়ের লেখাগুলোতে চোখ বুলিয়ে বইয়ের ব্যাপারে চমৎকার মতামত এবং পরামর্শ দেওয়ার জন্য। যারা আমার অনুপ্রেরণা, যাদের আমি গুরু মানি, তাদের কাছ থেকে এমন উৎসাহমূলক মতামত সত্যিই আমার জন্য অত্যন্ত আনন্দের।

বইয়ের সম্পাদনা এবং প্রুফ রিডে আন্তরিকভাবে সময় দিয়েছেন হাসান আব্দুল্লাহ, মুনতাসির বিল্লাহ মুন্না, আহমেদ ইসমাঈল হোসেন এবং তুর্জয় আব্দুল্লাহ ভাই। তাদের প্রতিও আমার অন্তরের অন্তঃস্তল থেকে কৃতজ্ঞতা।

বিশেষ কৃতজ্ঞতা অদম্য প্রকাশের স্বত্বাধিকারী নাজিব রাফে ভাইয়ের প্রতি। তার উৎসাহ এবং অনুপ্রেরণা ছাড়া এই বইটি হয়তো কখনোই আলোর মুখ দেখত না।

এ ছাড়াও যাদের কথা উল্লেখ না করলেই নয় তারা হলো আমার সহধর্মিণী, আমার আদরের কন্যা এবং আমার পরিবারের সদস্যরা। তাদের যেই সময়টুকু আমার দেওয়ার কথা ছিল, সেখান থেকে তারা ছাড় দিয়েছে বলেই আমি আমার ফুলটাইম চাকরির পরেও এই শখের লেখালেখি ও অন্যান্য কাজ করতে পারি। তাদের প্রতি কৃতজ্ঞতা জানানোর ভাষা আমার জানা নেই।

শেষ করার আগে একটা বিষয় উল্লেখ না করলেই না। এই বইয়ের টপিকগুলো গণিতের সূত্রের মতো নয় যে এটাই একমাত্র সঠিক আর এর বাইরে বাকি সব ভুল। বরং এখানের লেখাগুলো আমার নিজের ব্যক্তিগত অভিজ্ঞতা, ইন্ডাস্ট্রির অভিজ্ঞদের আর্টিকেল, বই, ইন্টারভিউ থেকে শেখা এবং সর্বোপরি আমার নিজের আন্ডারস্ট্যান্ডিংয়ের ওপর ভিত্তি করে লেখা। আমি নিজে যেহেতু এখনো প্রতিদিনই শিখছি, তাই আমার এই লেখাগুলোর ব্যাপারে যে কারও ভিন্ন মত থাকতে পারে। ভিন্ন কোনো মত যদি আপনার কাছে উত্তম মনে হয় এবং আপনার কাজের ক্ষেত্রে আরও বেশি প্র্যাক্টিক্যাল মনে হয় নিঃসন্দেহে আপনি সেটাই ফলো করবেন। আমার মূল উদ্দেশ্য হচ্ছে এই টপিকগুলো নিয়ে আলোচনা হোক। যারা এগুলো জানতেন

না তারা এগুলো জানুক, আরও বেশি এগুলো নিয়ে ঘাঁটাঘাঁটি করুক। আমার এই লেখাগুলো থেকে কেউ যদি সামান্য কিছুও শিখতে পারে সেটাই আমার প্রাপ্তি।

পরিশেষে, বইয়ের যেকোনো ভুল আপনার চোখে পড়লে আমাকে ব্যক্তিগতভাবে জানানোর অনুরোধ রইল। বইয়ের পরবর্তী সংস্করণে আমরা সেটি ঠিক করার সর্বোচ্চ চেষ্টা করব ইনশা আল্লাহ।

আহমেদ শামীম হাসান
ahmed.shamim.hassan@gmail.com

সূচিপত্র


C++
JS
PHP
HTML
XD
CSS

এত এত টিউটোরিয়াল থাকার পরেও কেন আমরা শিখতে পারি না?

আপনি হয়তো খুব মনোযোগ দিয়ে একটা বই পড়ে শেষ করলেন। পড়া শেষ হলে আপনি কি বইটার টপিক সম্পর্কে 'শিখে' ফেললেন?

অথবা আপনি কয়েক ঘণ্টা ব্যয় করে কোনো একটা বিষয়ের টিউটোরিয়াল বা ভিডিও দেখলেন। দেখা শেষ হলে ওই বিষয়ে আপনার কতটুকু 'শেখা' হলো?

'শেখা' বলতে আমরা আসলে কী বুঝি?

খুব সহজভাবে বলতে গেলে, কোনো একটা বিষয় আপনি যদি অন্তত কাজে লাগানোর মতো করে বোঝেন, তাহলে বলা যায় যে আপনি জিনিসটা শিখেছেন। ব্যাপারটা যদি কেবল থিওরেটিক্যাল বা কন্সেপচুয়াল হয় তাহলে আপনি যদি সেটা আরেকজনকে সহজভাবে বোঝাতে পারেন তাহলেও বলা চলে যে আপনি ব্যাপারটা শিখেছেন।

কিংবদন্তি জাতীয় অধ্যাপক আব্দুর রাজ্জাক স্যার বই পড়ে শেখার ক্ষেত্রে একটা চমৎকার পরামর্শ দিতেন, যেটা লেখক আহমদ ছফা তার 'যদ্যপি আমার গুরু' বইতে উল্লেখ করেছেন। তিনি বলতেন, একটা বই পড়ার পরে যদি সহজভাবে অল্প কথায় বইটার সারমর্ম বলতে না পারেন, তাহলে আপনি বইটা পড়ে কিছুই শেখেননি।

বিখ্যাত পদার্থবিদ Richard Feynman এরও এ রকম একটা উক্তি আছে,

> *If you cannot explain something in simple terms, you don't understand it.*

তার মানে শেখার পরে সেটাকে কাজে লাগানো বা আরেকজনকে বোঝাতে যাওয়াটাই আপনার জন্য লিটমাস টেস্ট যে আপনি আসলে কতটুকু শিখেছেন।

আমাদের শেখার ক্ষেত্রে সবচেয়ে বড় সমস্যা হলো, শেখার পেছনে আমরা যেই এফোর্টটা দিই, আমরা সেটা নিয়েই আত্মতৃপ্তিতে ভুগি। আসলে কতটুকু শিখলাম সেটা তখন আমাদের কাছে মুখ্য থাকে না। যেমন অনেক বাবা-মাই তাদের সন্তানদের বেশি বেশি পড়াশোনা করার জন্য চাপ দেন। বাইরে খেলতে যেতে দেন না, সামাজিক কোনো কাজেও অংশ নিতে দেন না। তাদের ধারণা বেশি পড়লেই সে বেশি শিখবে। কিন্তু আসলেই কি তাই? দেখা গেল তার চেয়ে অনেক কম এফোর্ট দিয়েও আরেকজন ভালো রেজাল্ট করছে।

আবার আমরা কতটুকু শিখলাম, সেটা কীভাবে মূল্যায়ন করা হচ্ছে সেটাও আসলে গুরুত্বপূর্ণ। যেই ছেলে/মেয়েটা রাত জেগে জেগে পরীক্ষার প্রিপারেশন নিয়ে গৎবাঁধা কিছু জিনিস মুখস্থ করে পরীক্ষায় ভালো রেজাল্ট করে এলো, তার কাছে মনে হতেই পারে সে অনেক কিছু পারে। ইংরেজিতে A+ পেয়েও যদি সে ইংরেজিতে একটা বাক্য রচনা করতে না পারে, সেটা তার জন্য কতটুকু সমস্যা এটা বুঝতে বুঝতে তার অনেক দেরি হয়ে যায়।

'Learning how to learn' অর্থাৎ কীভাবে শিখতে হয়, এটাও যেমন আমাদের শেখানো হয় না। আবার প্রকৃত অর্থে শেখা বলতে কী বোঝায় সেই ব্যাপারটা নিয়েও আমাদের মধ্যে ভুল ধারণা রয়েছে। এই টপিকে ইংরেজিতে বেশ ভালো কিছু কোর্স এবং রিসোর্স আছে। আগ্রহী হলে সেগুলো নিয়ে একটু ঘাঁটাঘাঁটি করা যেতে পারে।

যাই হোক, পুরো শিক্ষাব্যবস্থার সমস্যা নিয়ে আলাপ করে আমাদের লাভ নেই। আমাদের আলোচনা আপাতত সফটওয়্যার ইঞ্জিনিয়ারিং সেক্টরেই সীমাবদ্ধ থাকুক।

একটা সময় ছিল যখন প্রোগ্রামিং বা সফটওয়্যার ইঞ্জিনিয়ারিং শেখার একমাত্র রিসোর্স ছিল একাডেমিক কিছু বই। কিন্তু এখন প্রতিটা বিষয়ের অসংখ্য টিউটোরিয়াল, ডকুমেন্টেশন, আর্টিকেল, বই ইত্যাদি ইন্টারনেটে পাওয়া যায়। সহজ কথায় রিসোর্সের অভাব নেই। তারপরও নতুন কিছু শেখাটা আমাদের অনেকের জন্যই, বিশেষ করে এই সেক্টরে যারা নতুন বা জুনিয়র তাদের জন্য বেশ চ্যালেঞ্জিং।

এর প্রধানতম কারণ আমরা আসলে নিজেদের ইম্প্রুভ করতে চাই না।

শুনতে একটু অবাক লাগতে পারে যে, 'ইম্প্রুভ করতে চাই না' এ আবার কেমন কথা? সবাই-ই তো চায় নিজের ফিল্ডে আরেক্টু ভালো করতে, আরও এগিয়ে যেতে।

একটা উদাহরণ দিই। মনে করুন আপনি কোনো একটা ক্লাসের সব ছাত্রছাত্রীকে একত্র করে একটি সেশনের আয়োজন করলেন। সেখানে খুবই ট্যালেন্টেড এবং ভালো

রেজাল্ট করা একজন বড়ভাইকে নিয়ে এলেন যে কিনা তাদের ভালো রেজাল্ট করার গাইডলাইন দিতে পারবে। এখন যদি সেখানে জিজ্ঞেস করা হয়, কারা কারা ভালো রেজাল্ট করতে চায়, সবাই-ই বলবে যে তারা ভালো রেজাল্ট করতে চায়, তাই না? কিন্তু তাদের গাইডলাইন দেওয়া হলে সবাই-ই কি সেই গাইডলাইন ঠিকভাবে ফলো করবে? করবে না। কারণ মুখে বললেও সব ছাত্রছাত্রীর আসলে পড়াশোনার প্রতি তেমন একটা আগ্রহ থাকে না।

যে বিষয়ে আপনার আগ্রহ কম এবং ভালো করার ব্যাপারে আপনার ডিটারমিনেশন নেই, সে বিষয়ে আপনাকে যতই ট্রেনিং, গাইডলাইন দেওয়া হোক, আপনি ভালো করতে পারবেন না।

'Understanding software' বইতে লেখক এ জন্যই বলেছেন,

> *In order to become an excellent programmer, you must first want to become an excellent programmer. No amount of training will turn somebody who does not want to be excellent into an 'excellent programmer'.*

সুতরাং, প্রথম কাজ হচ্ছে 'শেখা' এবং 'ভালো করার' ব্যাপারে আপনার মনস্থির করা। দৃঢ়সংকল্প হওয়া।

রিসোর্স থাকার পরও ভালো করতে না পারার আরেকটি অন্যতম কারণ আমরা এগুলোকে সঠিকভাবে কাজে লাগানোর টেকনিক জানি না।

এই সমস্যা থেকে কীভাবে উত্তরণ করা যায় সে বিষয়েই কিছু আইডিয়া আমি শেয়ার করছি।

কোনো নতুন একটা টুল, ফ্রেমওয়ার্ক বা ল্যাঙ্গুয়েজ শেখা শুরু করলে নিচের স্টেপগুলো আপনাকে সাহায্য করবেঃ

- প্রথমেই সেটার খুঁটিনাটিতে বেশি ফোকাস না করা ভালো। প্রথমে উচিত একটা হাই-লেভেল আইডিয়া নেওয়া। জিনিসটা কী কাজ করে, এটা দিয়ে কী কী করা যায়, এর কী কী ফাংশনালিটি আছে এসব।

- মোটামুটি একটা ধারণা হয়ে গেলে সেটা নিয়ে টুকটাক কাজ করার ট্রাই করা উচিত। ইউটিউবে ঘণ্টার পর ঘণ্টা সাঁতার কাটা বা সাইকেল চালানোর টেকনিকের ওপর ভিডিও দেখে কি আপনি এই স্কিলগুলো আয়ত্ত করতে পারবেন? আপনাকে প্র্যাকটিস করেই এটা শিখতে হবে, তাই না? প্রোগ্রামিং বা এই রিলেটেড কিছু শেখার ক্ষেত্রেও একই ব্যাপার। আপনি হয়তো কোনো একটা টিউটোরিয়াল দেখে একটা প্রজেক্ট কীভাবে করে আইডিয়া নিলেন। কিন্তু আপনি নিজে কোনো কাজ করলেন না। তাহলে কি আপনার শেখা হলো? তার মানে আপনাকে হাতে-কীবোর্ডে কাজে নেমে পড়তে হবে।

- টিউটোরিয়াল দেখে প্রজেক্ট করতে করতে অনেকে টিউটোরিয়ালে যেভাবে দেখানো হচ্ছে হুবহু সেটাই জাস্ট কপি করে করে কোড করে রান করে দেখে। এটাও ভুল প্র্যাকটিস। এভাবে করে আপনি শিখতে পারবেন না। আপনার যেটা করা উচিত সেটা হলো, টিউটোরিয়ালটা একটানে দেখে শেষ করে পুরো ব্যাপারটার একটা আইডিয়া নেওয়া। প্রয়োজনে কিছু নোট নেওয়া। এরপরে টিউটোরিয়াল বন্ধ করে শুরু থেকে নিজের মতো করে কাছাকাছি একটা প্রজেক্ট আইডিয়া নিয়ে কাজ করা। এটা করতে গিয়ে হয়তো কিছু জায়গায় সমস্যা হবে, আটকে যাবেন - তখন স্পেসিসিক সেই সমস্যাটা নিয়ে ডকুমেন্টেশন দেখবেন বা সমাধান সার্চ করবেন। এভাবে করতে করতে জিনিসটা সম্পর্কে আপনার ধারণা আরও ক্লিয়ার হবে।

- আপনি যখন মোটামুটি কনফিডেন্ট এটা নিয়ে, তখন এটার খুঁটিনাটি, এটা পেটের ভেতরে কী আছে, কীভাবে এটা বানানো হলো এসব একটু আধটু দেখলে আপনি এটার ওপর মাস্টার হয়ে যাবেন। যেমন আপনি হয়তো একটা ফ্রেমওয়ার্কের ORM ব্যবহার করে কাজ করছেন। কিন্তু এটা বিহাইন্ড-দ্য-সিন কীভাবে একটা স্টেটমেন্টকে SQL এ কনভার্ট করছে, কীভাবে রিলেশনশিপগুলো হ্যান্ডেল করছে - এই ব্যাপারগুলোও আপনার জানতে হবে। ইন-ডেপথ ঘাঁটাঘাঁটি না করে আপনি যদি শুধু প্রজেক্টের পর প্রজেক্ট করে যান তাহলেও কিন্তু হবে না। কারণ আপনি এটা নিয়ে কাজ করতে গিয়ে কোনো জটিল সমস্যায় পড়লে সহজে সমাধান করতে পারবেন কেবল আপনার যদি এই খুঁটিনাটিগুলো সম্পর্কে ভালো ধারণা থাকে। অথবা কখনো যদি এমন হয় যে এটাকে ভেতর থেকে কাস্টোমাইজ করতে হবে বা অন্য কোনো টুল, ফ্রেমওয়ার্কের সঙ্গে এটাকে কম্বাইন করে কাজ করতে হবে, তখনো আপনার এই ইন-ডেপথ নলেজ আপনাকে হেল্প করবে।

এর বাইরে আরও কিছু বিষয় আপনার শেখাটা পাকাপোক্ত করতে সাহায্য করবেঃ

- আপনি যেটা শিখছেন সেটা অন্যদের শেখানো বা বই, আর্টিকেল, ভিডিও ইত্যাদির মাধ্যমে কম্যুনিটিতে আপনার নলেজ শেয়ার করতে পারেন। এতে করে আপনার নিজের বোঝাটা যেমন আরও সলিড হবে, একইসঙ্গে আপনার বোঝায় কোনো ভুল থাকলে সেটাও অন্যদের সামনে এক্সপোজ হবে। তাতে করে ফিডব্যাক থেকে আপনি নিজেকে ঠিক করে নিতে পারবেন।

- কোনো একটা জিনিস শিখে আপনার কাছে হয়তো এটাকেই বেস্ট মনে হচ্ছে। কিন্তু অভিজ্ঞ কারও সঙ্গে কথা বললে তিনি হয়তো আপনাকে আরও ভালো আইডিয়া দিতে পারবেন। এটার সুবিধা অসুবিধা এবং একই রকম অন্য আর কী অলটারনেটিভ আছে, সেগুলোর সুবিধা, অসুবিধা কী - এসব বিষয়ে। আবার নিজের ভুল যেহেতু নিজে ধরা যায় না, তাই দেখা যাবে টিউটোরিয়াল বা বিভিন্ন রিসোর্স থেকে আপনি কোনো একটা জিনিস ভুল শিখেছেন বা বুঝতে ভুল হয়েছে সেটা আপনি নিজে নিজে যাচাই করতে পারবেন না। এ জন্য সিনিয়র এবং অভিজ্ঞ কোনো একজন মেন্টরের তত্ত্বাবধানে থাকাটাও আপনার জন্য উপকারী হবে।

- রিলেটেড ওপেন-সোর্স প্রজেক্টগুলোতে কন্ট্রিবিউট করার চেষ্টা করাটাও শিখতে অনেক সাহায্য করবে। কারণ আপনি সেখানে ইন্ডাস্ট্রির এক্সপার্ট যারা আছে তারা কীভাবে কাজ করছে, কীভাবে একটা সমস্যা সমাধান করছে ইত্যাদি বিষয় শিখতে পারবেন।

শিখতে গিয়ে আমরা কিছু কমন ভুল করিঃ

- একসঙ্গে অনেক কিছু শিখতে যাই। এতে করে আপনার ফোকাস নষ্ট হয়। একটা বিষয়ে মনোযোগের সঙ্গে একটা লম্বা সময় আপনি যখন ঘাঁটাঘাঁটি করবেন তখন সেই বিষয়টা সম্পর্কে আপনার ভালো একটা আন্ডারস্ট্যান্ডিং তৈরি হবে। একেক সময় একেকটা নিয়ে ঘাঁটাঘাঁটি করলে সেটা হবে না। একটা বিষয়ে আপনি একটু কনফিডেন্ট হওয়ার পরে অন্য টপিকে মনোযোগ দেওয়া উচিত।

- শিখতে গিয়ে আমরা অনেক ক্ষেত্রেই 'সহজ টপিকে' ঘুরপাক খাই। নতুন, চ্যালেঞ্জিং কিছু শিখতে গেলে ব্রেইনের এফোর্ট বেশি দিতে হয়। যার কারণে ব্রেইন আমাদের ট্রিক করে সহজ টপিকে আটকে থাকতে। যেই টপিকটা আপনি হয়তো অলরেডি পারেন বা বোঝেন সেটার টিউটোরিয়াল, রিসোর্স

দেখতে গেলে আপনার ব্রেইনের কষ্ট কম হবে। দেখার পরে আপনার মনে হবে আপনি সময়টা 'শিখতে কাজে লাগিয়েছেন'। কিন্তু আসলে এর চেয়ে আপনার আরেকটু কঠিন কোনো টপিকে সময় দেওয়া উচিত ছিল। আপনি যদি কঠিন কিছু শেখার এই চ্যালেঞ্জটা গ্রহণ করতে না পারেন, ব্রেইনকে কষ্ট করাতে রাজি না করাতে পারেন, তাহলে আপনার খুব একটা ইম্প্রুভমেন্ট হবে না।

- আমরা যখন কোনো একটা কিছু ভুল করি, এটা আমাদের জন্য খুব ভালো একটা লার্নিং অপরচুনিটি। কিন্তু আমরা অনেকেই এই সুযোগটা কাজে লাগাই না। যখন কোনো একটা কিছু ভুল হচ্ছে, আপনি হয়তো এখান সেখান থেকে কোড বা সল্যুশন কপি করে ইস্যুটা সলভ করলেন। কিন্তু একইসঙ্গে আপনার শেখা উচিত কী কারণে এই সমস্যাটা হলো এবং পরবর্তী সময়ে এই সমস্যা যাতে আর না হয় তার জন্য কী করা লাগবে। এই অভ্যাসটা আপনাকে আপনার ক্যারিয়ারে অনেক দূর এগিয়ে যেতে সাহায্য করবে।

আশা করি এই টেকনিকগুলো কিছুটা হলেও আপনাকে সাহায্য করবে। তবে শেখার ক্ষেত্রে এগুলোই একমাত্র পদ্ধতি না। আপনি আপনার মতো করে ভিন্ন কোনোভাবেও শিখতে পারেন। শেখাটাই হলো মূল ব্যাপার, সেটা আপনি যেই টেকনিক কাজে লাগিয়েই শেখেন না কেন।

Imposter Syndrome

কখনো কি আপনার মনে হয়েছে যে আপনার আশপাশের সবাই খুব ট্যালেন্টেড, কেবল আপনিই কিছু পারেন না? হয়তো আপনি একটা ভালো কোম্পানিতে ভালো পজিশনে জব করছেন, ভালো পারফর্মও করছেন। এরপরও আপনার মধ্যে একটা হীনম্মন্যতা কাজ করে। মনে হয় অন্যরা আপনার চেয়েও অনেক ভালো পারফর্ম করছে। ঝটপট চমৎকার সব কাজ করে ফেলছে। কিন্তু আপনার কোনো একটা কাজ করতে অনেক সময় লেগে যাচ্ছে। অথবা হয়তো এমন মনে হয় আপনি যে আসলে এটা, ওটা, অনেক কিছুই পারেন না - এই ব্যাপারটা অন্যরা জেনে গেলে আপনার মানসম্মান থাকবে না।

এটাই হচ্ছে 'Imposter syndrome'. নিজের সম্পর্কে হীনম্মন্যতা, নিজের জানার ঘাটতি অন্যদের সামনে প্রকাশিত হয়ে যাওয়ার ভয় এবং আত্মবিশ্বাসের অভাব। দেখা যায় আপনার বলার মতো যথেষ্ট যোগ্যতা, সাফল্য থাকার পরেও আপনি অন্যদের আপনার থেকে বেশি যোগ্য, সফল মনে করে তাদের সঙ্গে নিজেকে বারবার তুলনা করছেন এবং তাদের থেকে পিছিয়ে আছেন - এমন মনে করছেন।

Image source: https://workchronicles.com/

এই চিন্তার সবচেয়ে খারাপ দিক হলো, আপনি আরও ভালো কিছু করার, চ্যালেঞ্জ নেওয়ার মোটিভেশন হারিয়ে ফেলবেন। নিজের মধ্যে সব সময়ই একটা ভয়, হতাশা কাজ করবে। অথচ আপনার মধ্যে যেই যোগ্যতা এবং পটেনশিয়াল আছে সেটাকে কাজে লাগিয়ে আপনি হয়তো অনেক যুগান্তকারী কাজ করে ফেলতে পারতেন। অথবা ইতোমধ্যেই হয়তো আপনি অনেক চমৎকার কাজ করেছেন, কিন্তু সেগুলোকে কেবল 'ভাগ্যগুণে' পাওয়া মনে করছেন। নিজের যোগ্যতার স্বীকৃতি দিচ্ছেন না।

Imposter syndrome বেশি দেখা যায় জুনিয়র বা যারা মাত্র জবে ঢুকেছে তাদের মধ্যে। ভার্সিটি থেকে পড়াশোনা করার পর যখন রিয়েল ওয়ার্ল্ডে কাজ করতে নামা হয় তখন মনে হয় যে, আমি তো কিছুই পারি না। আমি তো তেমন কোনো কন্ট্রিবিউটই করতে পারছি না। আমি যে কিছু পারছি না এতে মনে হয় অন্যরাও আমার ওপর বিরক্ত হচ্ছে। এসব চিন্তার কারণে তারা কোনো কিছু না বুঝলে প্রশ্ন করতে বা কাউকে জিজ্ঞেস করে জেনে নিতেও ইতস্ততবোধ করে। চ্যালেঞ্জিং কোনো কাজ নিতে ভয় পায়। তার করা কোডে রিভিউ করে কোনো ভুল ধরিয়ে দিলে সে লজ্জাবোধ করে। সব সময় একটা মানসিক প্রেশারে থাকে যে তাকে আরও ভালো করতে হবে, নাহলে হয়তো তাকে কোম্পানি বাদ দিয়ে দেবে।

কিন্তু বাস্তবতা হচ্ছে এই ব্যাপারটা কেবল জুনিয়রদের মধ্যে দেখা যায় এমন না। অনেক সিনিয়রও এ রকম ফিল করেন। এমনিতেই যেহেতু টেকনোলজি সেক্টরে প্রতিনিয়ত নিত্যনতুন জিনিস আসছে, তাই এগুলোর সঙ্গে নিজেদের মানিয়ে নেওয়ার একটা তাগাদা সবার মধ্যেই থাকে। এর মধ্যে যখন মনে হয় আশপাশের সবাই নতুন নতুন জিনিস শিখে এগিয়ে যাচ্ছে, তখন জুনিয়র-সিনিয়র সব ধরনের সফটওয়্যার ইঞ্জিনিয়াররাই কখনো না কখনো এই Imposter syndrome-এ আক্রান্ত হন।

সমস্যা হচ্ছে এটার সহজ কোনো সমাধান নেই। অনেকের কাছে মনে হয় আরও বেশি জানলে, আরও বেশি শিখলে হয়তো আমি Imposter syndrome থেকে মুক্তি পাব। কিন্তু মজার ব্যাপার হচ্ছে, আপনি যত বেশি শিখবেন তত বেশি জানবেন যে আপনি আরও অনেক কিছুই জানেন না। তাই আরও বেশি জানাটা এটার সমাধান না। বরং এটা যেহেতু সাইকোলজিক্যাল ব্যাপার, তাই দৃষ্টিভঙ্গির পরিবর্তনই কেবল পারে এই সমস্যা থেকে উত্তরণে সাহায্য করতে।

সবচেয়ে প্রথমে যেই কথাটা মাথায় রাখতে হবে তা হলো, দুনিয়ার কেউই 'সব' কিছু পারে না। সবারই কিছু না কিছু জানার ঘাটতি আছে। আপনি যাকে দেখে মনে করছেন যে সে অনেক ট্যালেন্টেড, সব পারে। আসলে সে-ও অনেক কিছু পারে

না। সুতরাং আপনাকেও সবকিছু পারতে হবে এমন না। আপনি নিজের ফোকাস এবং টার্গেট রাখবেন কিছু নির্দিষ্ট বিষয়ের ওপর, যেটা কিনা আপনার ক্যারিয়ারের প্রাইমারি গোলের সঙ্গে রিলেটেড। সেগুলোতেই আপনি নিজেকে আরও বেটার করার চেষ্টা করবেন।

আমরা সবচেয়ে বেশি হীনম্মন্যতায় ভুগি আরেকজনের সঙ্গে নিজেকে তুলনা করে। আমরা যখন আরেকজনকে ভালো কিছু করতে দেখছি, তখন তার থেকে আমাদের উৎসাহিত হওয়া উচিত। তাকে দেখে নিজেকে ছোট মনে করা উচিত না। চিন্তা করা উচিত, সে যখন পেরেছে তাহলে আমিও পারব।

আপনি হয়তো ১০ মিনিটের কোনো একটা ভিডিও টিউটোরিয়াল দেখে চমৎকৃত হয়ে ভাবছেন, ওয়াও, এই লোক কত ট্যালেন্টেড। কী রকম ঝটপট করে, নিখুঁতভাবে এই রকম একটা কাজ করে ফেলছে। অথচ আপনি জানেন কি, এই ১০ মিনিটের ভিডিও বানাতে গিয়ে তার হয়তো ৪-৫ ঘণ্টা লেগেছে? আপনি দেখছেন তার পোলিশ করা, এডিট করা সুন্দর একটা কাজ। কিন্তু এর পেছনে তার ভুলগুলো বা এটা বানানোর জন্য তার পড়াশোনা, প্রিপারেশন কিছুই আপনি এই ফাইনাল আউটপুটে দেখছেন না। সুতরাং মনে করার কোনো কারণ নেই যে সে একেবারে নিখুঁত কাজ করছে। সবারই ভুল হয়। সবাইই কাজ করতে করতে শেখে। এক্সপার্টরাও সময়ে সময়ে গুগলের সাহায্য নেয়, অন্যদের থেকে শেখে।

কোনো কিছু না জানাটা কারও দোষ না। বরং এই না জানার ব্যাপারটা স্বীকার না করা বা জানার চেষ্টা না করাটাই সমস্যা। তাই কোনো বিষয়ে আপনার জানা না থাকলে সেটা মেনে নিন এবং জানার চেষ্টা করুন।

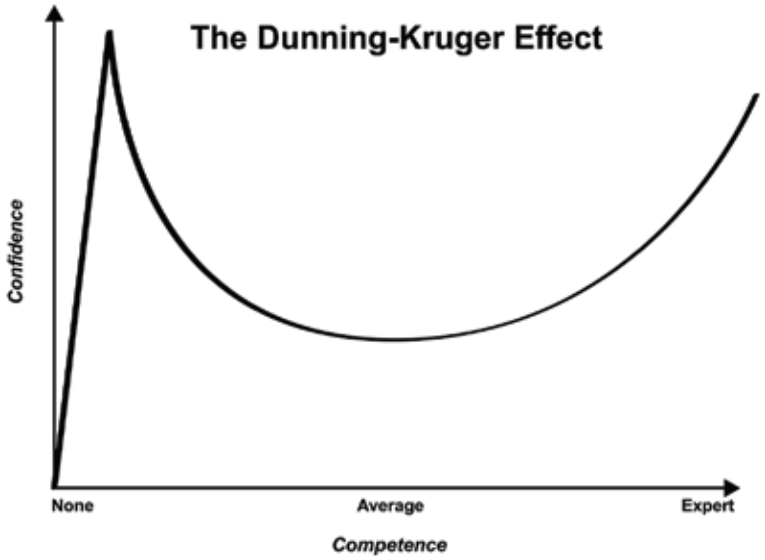
এই বিষয়গুলো মাথায় রাখলেই Imposter syndrome থেকে অনেকখানি মুক্তি পাওয়া সম্ভব।

আপনি যদি সিনিয়র ডেভেলপার হন এবং দেখেন যে আপনার কোনো জুনিয়র বা আশপাশের কেউ Imposter syndrome-এর কারণে স্ট্রাগল করছে, তাহলে আপনার উচিত হবে তাকে সাহায্য করা। তাকে আশ্বস্ত করা, উৎসাহ দেওয়া, তার অ্যাচিভমেন্টগুলোকে হাইলাইট করা এবং তাকে পজিটিভ ফিডব্যাক দিয়ে ইম্প্রুভ করার জায়গাগুলোতে গাইড করা।

Imposter syndrome-এর সঙ্গে প্রাসঙ্গিকভাবে আরেকটা টপিক চলে আসে যেটা কিনা এর বিপরীত। সেটা হলো Dunning-Kruger effect। আমরা এমন অনেককেই দেখি যারা কোনো একটা বিষয়ে সামান্য ধারণা নিয়েই মনে করে সে সেই বিষয়ে এক্সপার্ট

হয়ে গেছে। এই যে নিজের সম্পর্কে তার ওভারকনফিডেন্স এবং অতি উচ্চ ধারণা, যার কোনো বাস্তবতা নেই, এটাকেই বলা হয় Dunning-Kruger effect।

এ ব্যাপারটাও জুনিয়রদের মধ্যে বেশি দেখা যায়। দেখা যায় কেউ একজন প্রোগ্রামিং শেখা শুরু করার কিছুদিন পরেই মনে করে সে অনেক কিছু পারে। সে তখন যা পারে সেটা নিয়ে অন্যদের সঙ্গে তর্ক জুড়ে দেয়। সোশ্যাল মিডিয়াতে কেন অমুক প্রোগ্রামিং ল্যাঙ্গুয়েজ ভালো, কেন তমুক ফ্রেমওয়ার্ক খারাপ এগুলো নিয়ে বাগবিতণ্ডায় লিপ্ত হয়। অথচ সে এখনো এই বিষয়গুলো সম্পর্কে বিস্তর ধারণাই রাখে না।



সময়ের সঙ্গে সঙ্গে যখন সে অন্যান্য বিষয়ে আরও বেশি জানে, শেখে, তখন সে বুঝতে পারে নিজের ভুলগুলো। যদি সে সচেতন হয় এবং ওপেন-মাইন্ডেড হয়, তাহলে সে নিজেকে শুধরে নিতে পারে।

যাই হোক, এই দুই ধরনের মানসিক অবস্থা সম্পর্কেই সচেতন থাকা দরকার। কারণ এ দুটোই আমাদের যোগ্যতা এবং দক্ষতা উন্নয়নের ক্ষেত্রে বাধা হয়ে দাঁড়াতে পারে।