

**stack কথা**

# Lifecycle of a Full Stack Developer



**HM Nayem**

# **Lifecycle of a Full Stack Developer**

**- HM Nayem**



Copyright © 2020 Stack Learner

এই বইয়ের সমস্ত বিষয় বস্তু Stack Learner এবং HM Nayem এর কাছে সংরক্ষিত। এই বইয়ের কোনো লেখা বিনা অনুমতিতে প্রিন্ট করা বা নিজের নামে চালানো যাবে না। এই বইয়ের সমস্ত বিষয় বস্তু লেখকের নিজস্ব চিহ্ন ভাবনা থেকে নেওয়া। তাই কোনো কিছু গ্রহণ করার পূর্বে অবশ্যই নিজ দায়িত্বে তার গ্রহণযোগ্যতা যাচাই করে নিবেন। এই বইয়ে ব্যবহৃত টেকনোলজি এবং রেফারেন্স লিংক গুলো ইন্টার্নেট রিসার্চের মাধ্যমে নেওয়া। কোনো প্রকার কোনো কোম্পানির অ্যাফিলিয়েশন এখানে করা হয় নি। তাই এই বইয়ের সাথে দেওয়া সমস্ত রেফারেন্স লিংক নিজে যাচাই করেই ব্যবহার করবেন। এই বইয়ের কোনো অংশ আপনার ওয়েবসাইটে অথবা অন্য কোনো ক্ষেত্রে ব্যবহারের জন্য Stack Learner অথবা লেখকের অনুমতির প্রয়োজন হবে।

বইয়ের বিষয়বস্তু এবং ধারণা

HM Nayem

Shegufa Taranjum

Shayaike Salvy

বইয়ের প্রচ্ছদ

Shayaike Salvy

বই ডিজাইন

HM Nayem

অন্যান্য -

Shegufa Taranjum

Shayaike Salvy

Tawhidi Bari

Md Monirul Islam

Asief Mahir

Saidur Rahman Setu

# উৎসর্গ

বাংলাদেশের সকল সফটওয়্যার সৈনিক যারা দেশের জন্য নিজেকে সফটওয়্যার যোদ্ধা বানাতে প্রস্তুত

# সূচিপত্র

- |  |   |
|--|---|
| <p><b>১</b> শুরুর কথা ০৬</p> <p><b>৩</b> ভাষার ব্যাকরণ ৫০</p> <p><b>৫</b> শিল্প না সাহিত্য ৮৩</p> <p><b>৭</b> সাহিত্যের সৌন্দর্য ১২৩</p> <p><b>৯</b> মেরুদণ্ডের কাঠামো ১৫৫</p> | <p><b>২</b> বোকা বাক্যের ভাষা ১৮</p> <p><b>৪</b> সাহিত্য যাত্রা ৬৫</p> <p><b>৥</b> সাহিত্যের ভাষা ১০৮</p> <p><b>৮</b> সাহিত্যের মেরুদণ্ড ১৩৪</p> <p><b>১০</b> শেষের কথা ১৬৮</p> |
|--|---|



---

অধ্যায় এক

# শুকুর কথা

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ ডেভেলপমেন্ট শুরুর কথা
- ✓ কম্পিউটার সাইন্সের গুরুত্ব
- ✓ প্রোগ্রামিং ল্যাংগুয়েজের গুরুত্ব

---

যুগে যুগে মানুষ জীবিকা নির্বাহের জন্য কত কিছুই না করেছে। এখনো করছে, ভবিষ্যতেও করবে। তবে এখন মানুষ ঘরে বসেই অর্থ উপার্জন করতে পারে। সামনে শুধু একটা বোকা বাক্তা থাকলেই হয়। সারা দিন বোকা বাক্ত্বের সামনে বসে কিবোর্ডে টুন টুন আওয়াজ করলেই লক্ষ লক্ষ টাকা চলে আসে। এই রকম একটা ধারণা থেকেই বেশির ভাগ মানুষ, নিজের অজ্ঞতাই একটা গহীন জংগলে পদার্পণ করে। যেমনটা আমিও করেছিলাম, নয় বছর পূর্বে।

জংগলটা যে এত ঘন আর অন্ধকার, সেটা শুরুর দিকে বুঝতে পারা যায় না। আপনি যতই এই জংগলের ভিতরে প্রবেশ করতে থাকবেন, ততই মনে হবে যেন হারিয়ে যাচ্ছেন। পিছনে ফিরে আসার কোনো অবস্থাও আর থাকবে না। কারণ জংগলে হাঁটতে আপনার এত ভাল লাগবে যে আপনি আর পিছনে ফিরে যেতে চাইবেন না। আবার চোখের সামনে যখন জীবন্ত একটা স্বপ্ন ঝুলবে তখন মনে হবে, আর একটু যাইনা। আর একটু গেলেই হয়ত আমার কাঞ্চিত লক্ষ্যে পৌঁছে যাব।

আপনি শুধু হেঁটেই যাবেন, হেঁটেই যাবেন। কোনো দিক নির্দেশনা নেই, কোনো সঠিক তথ্য নেই, যে আর কতটা দোঁড়ালে নিজের গভৰ্বে পৌঁছানো যাবে। আপনি আপনার আশেপাশের মানুষজনের দিকে তাকাবেন। দেখবেন, তারা তো অনেক দূর এগিয়ে গিয়েছে। তারা এই পথে যাচ্ছে, তাই আমারও এই পথেই যাওয়া উচিত। এটা ভেবেই আপনি আবারও নতুন পথের উদ্দেশ্যে রওনা দিবেন, সেই পথ কতটা কাঁটাময় সেটা বিবেচনা না করেই। সেই পথ সম্পর্কে নৃন্যতম জ্ঞান না থাকার পরেও আপনি যাবেন এবং যাওয়ার পরে নতুন করে একটা হাঁচট খাবেন।

ফ্রিলান্সিং করে অনেক টাকা উপার্জন করা যায়, লক্ষ লক্ষ টাকা চলে আসে কীবোর্ডের কয়েকটা টুকাতে। সত্য গোপন করে এই রকম কথা আপনাকে বলার মানুষের অভাব আপনি পাবেন না। কিন্তু কেউ আপনাকে বলবে না, এই টাকা অর্জনের পূর্বে তাকে কি করতে হয়েছে। কত দিন, কত রাত, কত বছর ওই গহীন জংগলে একাকী তারার মিটি মিটি আলোয় নশ পায়ে পথ চলতে হয়েছে। এখনো যদি ভালো করে লক্ষ্য করেন, তাদের ক্ষতস্থান গুলো আপনার নজর এড়াবে না।

ফ্রিলান্সিং করে অনেক টাকা উপার্জন করা যায় এটা চিরন্তন সত্য। তবে আর একটা লুকায়িত সত্য হচ্ছে আপনাকে দক্ষ হতে হবে। এই দক্ষতার কোন মাপকার্তি নেই। আপনি যদি ডাক্তার

---

হতেন, তাহলে নির্দিষ্ট একটা পড়াশোনা শেষ করার পরে আপনাকে নির্দিষ্ট একটা কাজই করতে হতো। বা আপনি যদি একজন আর্কিটেক্ট হতেন তাহলেও পড়াশোনা শেষে একটা নির্দিষ্ট টাইপেরই কাজ আপনাকে করতে হতো। কিন্তু ফ্রিলান্সার হিসেবে আপনাকে ঠিক কখন কি কাজ করতে হবে, তা আপনি জানেন না। তাই ফ্রিলান্সার হিসেবে টিকে থাকতে হলে আপনাকে অনেক অনেক বেশি জ্ঞান অর্জন করতে হবে।

আমাদের দেশে ফ্রিলান্সিংটাকে সর্বশেষ পেশা হিসেবে ধরা হয়। যখন কোনো কিছুই আর পারছি না, তখন একটা কম্পিউটার কিনে শুধু বসে পরার অপেক্ষা, টাকা তো আমার জন্য রেডি হয়েই আছে। এই রকম মানুষের সংখ্যাও কিন্তু কম না। আমাদের দেশে ফ্রিলান্সিং করতে চাওয়া মানুষের সংখ্যা অগণিত। কিন্তু সত্যিকার অথেই নিজের দক্ষতা বৃদ্ধি করতে চাওয়া মানুষের সংখ্যা কিন্তু হাতে গুনে বের করা যাবে। এখন অবশ্য এই সংখ্যাটা পরিবর্তিত হচ্ছে। মানুষ শিখতে চাচ্ছে, নিজের জ্ঞান বৃদ্ধি করতে চাচ্ছে। কিন্তু যত মানুষ নিজেকে জ্ঞান গরিমায় সমৃদ্ধ করতে চাচ্ছে তার ১ ভাগ মানুষও নিজের জ্ঞানকে বিতরণ করতে চাচ্ছে না।

যার ফলে কি হচ্ছে, অন্ধকার ঘন জংগলটা অন্ধকারই থেকে যাচ্ছে। শেখার সময়টা বৃদ্ধি পাচ্ছে এবং যুগের সাথে তাল মিলিয়ে আর চলতে শেখা হচ্ছে না। পুরো দেশ, জাতি পিছিয়ে পরছে। এমন অবস্থায় আমাদের দরকার সঠিক দিকনির্দেশনা। এই দিক নির্দেশনা গুলো আপনারা যদি একটু ইন্টার্নেটে সার্চ করেন তাহলেই পেয়ে যাবেন। আপনার বক্স, বড়ভাই কে কি করছে তার দিকে তাকিয়ে থাকতে হবে না। নিজের ডিসিশন নিজেকেই নেওয়া শিখতে হবে। একটা কথা সব সময় মাথায় রাখবেন, ফ্রিলান্সিং মার্কেটপ্লেস গুলোতে বিক্রি হয় আপনার দক্ষতা, অন্য কিছু না। আর আপনি দক্ষ হলে, আপনার দক্ষতা আপনি অনেক ভাবেই বিক্রি করতে পারবেন।

প্রোগ্রামিং, ওয়েব ডেভেলপমেন্ট বা গেম ডেভেলপমেন্ট যায় বলেন না কেন, এগুলো হচ্ছে প্যাশনের জায়গা। এটা গুরুত্বপূর্ণ না যে প্যাশনটা আগে থেকেই ছিল নাকি এটা শিখতে এসে জন্ম নিয়েছে, গুরুত্বপূর্ণ বিষয় হল প্যাশনটা জন্ম নিয়েছে কিনা। কারণ জোর করে ঘণ্টার পর ঘণ্টা একটা বোকা বাক্সের সামনে বসে থাকা যায় না। একটা বোকা বাক্সে শিক্ষিত মানুষের ব্যবহার উপযোগী করা যায় না।

---

ডেভেলপমেন্ট হচ্ছে কম্পিউটারের জগতে সব থেকে কঠিন কাজ, ডেভেলপমেন্টের সাথে আরও অনেক কাজ জড়িত আছে কিন্তু আমি এক কথায় ডেভেলপমেন্টকেই ধরে নিচ্ছি, কারণ সব শেষে আমরা একটা ডেভেলপড প্রোডাক্টই পেয়ে থাকি। আমরা অনেকেই কম্পিউটারের সাথে ভালোভাবে পরিচিত হওয়ার আগেই আমাদের আশেপাশের মানুষজনের কথা শুনে, তাদের দ্বারা প্রভাবিত হয়ে ডেভেলপমেন্ট করতে নেমে পরি। প্রোগ্রামিং বলে যে দুনিয়াতে কিছু একটা আছে, সেই বিষয়টাও জানতে পারি ডেভেলপমেন্ট করতে এসেই। আসলে এটা মাটেও ঠিক না। আমাদেরকে সঠিক পদক্ষেপের ভিতর দিয়েই এগিয়ে যেতে হবে। তা না হলে কাজের কাজ কিছুই হবে না, শুধু সময়টায় নষ্ট হবে। সবার প্রথমে আমাদের কম্পিউটারের ব্যবহার ভালোভাবে শিখতে হবে। কম্পিউটার ভালো ভাবে ব্যবহার করতে না জানলে কোনো দিনও এর জন্য সফটওয়্যার বানানো সম্ভব না। ডেভেলপমেন্টের উদ্দেশ্যে কম্পিউটার কিনতে চাওয়া বা কম্পিউটার কিনেই, ভালো ভাবে এর ব্যবহার না শিখেই ডেভেলপমেন্ট করতে চাওয়া মানুষের সংখ্যা আমাদের দেশে লক্ষ্যাধিক।

আমরা কম্পিউটার সাইন্সের স্টুডেন্ট হই আর নাহই, ডেভেলপমেন্ট সেক্টরে টিকে থাকতে চাইলে কম্পিউটার সায়েন্স সম্পর্কে আমাদের নুন্যতম একটা জ্ঞান অর্জন করতে হবে। কেন করতে হবে সেটা বোঝার জন্য ছোট একটা উদাহরণই যথেষ্ট বলে আমি মনে করি। ইলেক্ট্রিক্যাল ইঞ্জিনিয়ার আর ইলেক্ট্রিক্যাল মিস্কি দুইজনই ইলেক্ট্রিক্যাল কাজ করে থাকে। কিন্তু একজনকে আমরা ইঞ্জিনিয়ার বলি, কারণ তার ইঞ্জিনিয়ারিং জ্ঞান রয়েছে। তিনি ইলেক্ট্রিক্যাল বিষয়ে দক্ষ একজন মানুষ, সমস্ত ক্রিটিক্যাল অবস্থার মোকাবেলা তিনি করতে পারেন। আর একজন কে আমরা মিস্কি বলি, কারণ তিনি কোন সূত্র জানেন না। তার ইলেক্ট্রিক্যাল বিষয়ে কোনো পুঁথিগত বিদ্যা নেই, তিনি দেখে দেখে শিখেছেন। আপনাদেরকে প্রথমেই সিদ্ধান্ত নিয়ে নিতে হবে যে আপনারা কি হবেন? আর একটা বিষয়ও এর সাথে আপনাকে মাথায় রাখতে হবে যে, কয়েক বছরের নিরলস পরিশ্রমের পর, কয়েক বছর নির্ঘূর্ম রাত্রি যাপন করার পরই একজন ইঞ্জিনিয়ার জন্ম নেয়।

ইন্টারনেটের জগতে কম্পিউটার সাইন্সের যত রিসোর্স রয়েছে তা আর কোন এডুকেশন ফিল্ডেই নেই। তাই আপনি কম্পিউটার সাইন্সের ছাত্র বা ছাত্রী না হলেও কোনো সমস্যা নেই। আপনি চাইলেই সব কিছু নিজে থেকেই ঘরে বসে বসেই শিখে নিতে পারবেন। তবে সেই ক্ষেত্রে আপনাকে হাতে সময় নিয়ে বসতে হবে। কম্পিউটার সাইন্স এমন কোন বিষয় না, যে একমাসে আপনি আয়ত্ত করে ফেলতে পারবেন। সারা জীবন যদি শেখার মানুষিকতা থাকে, শুধু মাত্র তাহলেই এই জগতে আপনি নিজের অবস্থান তৈরি করতে পারবেন।

---

অনেক নবীন ডেভেলপার প্রশ্ন তুলেন যে, আমি তো ওয়েব ডেভেলপমেন্ট করতে চাই, আমার বোধহয় কম্পিউটার সাইন্স শাখা লাগবে না। যদি এই রকম ভাবে চিন্তা করেই থাকেন তাহলে আমি আপনাকে বলবো, নতুন করে চিন্তা শুরু করুন। আগের চিন্তাটা ভুল ছিল। কারণ ডেভেলপমেন্ট, সেটা যে কোনো কিছু ডেভেলপ করা হতে পারে, কম্পিউটার সাইন্সের একটা শাখা মাত্র। যে কোনো কিছু ডেভেলপ করতে গেলেই আপনার কম্পিউটার সাইন্সের সূত্র গুলোই কাজে লাগবে। আর এই সূত্র গুলো না জেনে, না বুঝে যদি আপনি কোনো অ্যাপলিকেশন ডেভেলপ করেন তাহলে সেটা ইলেক্ট্রিক্যাল মিস্ট্রির মতই শুধু কিছু তারের কানেকশন হবে, যাতে কাজ চলবে কিন্তু এর বেশি কিছু সম্ভব না। একটা কথা মাথায় রাখবেন, একটা বিউল্ডিং তৈরিতে ইঞ্জিনিয়ার এবং মিস্ট্রি কাঁধে কাঁধ মিলিয়ে কাজ করে। ইঞ্জিনিয়ারের অনেক কাজ মিস্ট্রি করলেও সে ইঞ্জিনিয়ার হয়ে যায় না। আপনার চারপাশে তাকালেই আপনি ইঞ্জিনিয়ার এবং মিস্ট্রি এর পার্থক্যটা বুঝতে পারবেন। আর আমি চাই, আপনি ডিজিটাল মিস্ট্রি না হয়ে একজন ভালো ইঞ্জিনিয়ার হয়ে গড়ে উঠুন। শুধুমাত্র তাহলেই আমাদের দেশ শ্রম বির্কি না করে মেধা বির্কি করতে পারবে।

ডেভেলপমেন্ট জগতে আপনার কাজ বোকা বাক্তাকে নিজের গোলাম বানিয়ে স্বার্থ হাসিল করা, নিজের কাজ করিয়ে নেওয়া। আপনি এই বোকা বাক্তাকে যদি নিজের মত করে কন্ট্রোল করতে চান, সেটা ডেভেলপমেন্ট করেই হোক আর হ্যাক করেই হোক, আপনাকে প্রথমে এর ভাষা শিখতে হবে। আর তার পরে কম্পিউটার সাইন্সের থিওরি গুলো জানতে হবে। কম্পিউটার আসলে খুবই বোকা, এর নিজের কোনো কিছু করার ক্ষমতা নেই। আমরা ডেভেলপাররা এই বোকা বাক্তাকে বুদ্ধিমান বানিয়ে থাকি। তাহলে নিশ্চয় বুঝতে পারছেন আমাদেরকে কি পরিমাণ বুদ্ধিমান হতে হবে যদি এই বোকা বাক্তাকে বুদ্ধিমান বানাতে হয়।

আমি মনে করি, যদি আপনি কম্পিউটার ইঞ্জিনিয়ার হতে চান তাহলে আপনাকে কম্পিউটার হতে হবে। মানে আপনার ৱেইনকে কম্পিউটারের মত ফাস্ট কাজ করাতে হবে, কম্পিউটারের মত লজিক্যাল হতে হবে। নিজের ৱেইনে সব কিছু জমা করে রাখার দরকার নেই, যত টুকু দরকার শুধু ততটুকু জমা করে রাখলেই চলবে। আর নিজের মাথাকে সব সময় প্রসেসরের মত ক্যালকুলেশনের কাজে ব্যস্ত রাখতে হবে। কম্পিউটারের জগতটা এমননা যে একবার ডজনখানেক বই মুখ্য করে গুলিয়ে খেয়ে নিতে পারলেই হয়ে গেল। এবার সারা জীবন এই বই গুলোর নির্যাস বেঁচেই চলবে। এখানে প্রতিটা দিন, প্রতিটা মুহূর্ত আপনাকে নতুন নতুন সমস্যার সম্মুখীন হতে হবে। এই বোকা বাক্তাটা একটা করে নতুন চ্যালেঞ্জ আপনার দিকে ছুঁড়ে দিয়ে মুচকি মুচকি হাসবে। যা সমাধান করতে কোনো বই, কোনো টিউটোরিয়ালই কাজে

---

আসবে না, কাজে আসবে শুধু আপনার বুদ্ধিমত্তা। যত আপনি এর গভীরে প্রবেশ করতে থাকবেন প্রলেম গুলোর সাইজ, কম্প্লেক্সিটি ও তত বড় হতে থাকবে। আর তখনই কাজে আসবে কম্পিউটার সাইন্সের জ্ঞান।

তবে আমি মনে করি, কম্পিউটার সাইন্সের এই জগতে টিকে থাকার সব থেকে সহজ এবং সুন্দর উপায় হচ্ছে কম্পিউটারের সাথে বন্ধুত্ব করে নেওয়া। এটা একটা বোকা বাক্তা, একবার যদি আপনি এর সাথে বন্ধুত্ব করতে পারেন, তাহলে এ আপনার সব কথায় শুনবে। যেভাবে খুশি সেভাবেই নাচাতে পারবেন কম্পিউটারকে। তবে তার জন্য প্রথমে আপনাকে একটা কাজ করতে হবে, আর সেটা হচ্ছে কম্পিউটারের ভাষা শিখতে হবে। যেহেতু কম্পিউটার খুবই বোকা, তার তো ক্ষমতা নেই যে সে আপনার ভাষা শিখবে। তাই এটাই সব থেকে ভাল হয় যে, আপনি তার ভাষা শিখে ফেলুন। আর একবার তার ভাষা যদি আপনি রপ্ত করতে পারেন, তার ভাষায় আপনি তাকে যা করতে বলবেন সে কিন্তু তাই করবে, প্রদীপ থেকে বের হওয়া জীনের মতই আপনার সমস্ত হৃকুম মনে চলবে।

অনেকেই মনে করে থাকেন প্রোগ্রামিং খুব কঠিন। তাদের এমনটা ভাবা দোষের কিছু নয়। কারণ আমাদের আশেপাশের মানুষজন যারা এর সাথে যুক্ত আছেন, তারাই আমাদেরকে এমনটা ভাবতে বাধ্য করছে। এই ক্ষেত্রে আমি তাদেরকেও দোষ দেই না। কারণ ব্যাপারটা ইংরেজি এবং অংকের মত। এই দুইটা বিষয় খুব গুরুত্বপূর্ণ জীবনে, তাই ছোট বেলা থেকেই বাবা মা এর গুরুত্ব বোঝানোর জন্য অনেক রকম ভাবে চেষ্টা করে গেছেন। আর লাভের লাভ কি হয়েছে? ইংরেজি এবং অংক দুইটা বিষয়ই আমাদের ভয়ের কারণ হয়ে দাঁড়িয়েছে। প্রোগ্রামিং এর ক্ষেত্রেও ব্যাপারটা একই রকম। সবাই এর গুরুত্বটাই আমাদেরকে বোঝাতে চেয়েছে, কিন্তু আমরা বুঝেছি এটা খুবই কঠিন। আর কোন কিছুকে আগে থেকেই কঠিন ধরে নিয়ে বেশি দূর আগানো সন্তুষ্ট নয়। কিছুক্ষণ পর পর মনে হবে আমার দ্বারা সন্তুষ্ট না। আমি আমার নিজের লাইফেই অসংখ্যবার এই রকম সময়ের মুখোমুখি হয়েছি, যখন মনে হয়েছে প্রোগ্রামিং আমার জন্য না।

প্রোগ্রামিং কঠিন কিছু না, আবার একদম সহজ কিছুও না। সব থেকে ভাল হয় যদি আমরা তুলনা করায় ছেড়ে দেই। কঠিন হোক বা সহজ, কম্পিউটারের সাথে বন্ধুত্ব করতে চাইলে আমাদেরকে তার ভাষা শিখতেই হবে, ডিসিশন এখন আপনার। প্রোগ্রামিং বলতে এতক্ষণ মূলত আমি প্রোগ্রামিং ল্যাংগুয়েজ এর কথায় বোঝাচ্ছি। ব্যাপারটাকে এভাবে ধরে নিতে পারেন যে, আপনি সারা জীবনের জন্য জার্মানি অথবা স্পেনে চলে যাবেন। তাহলে এখনতো

---

তাদের ভাষা শিখতে হবে, তাই না? নাহলে তো সব রকম সুযোগ সুবিধা আপনি পাবেন না। যেমন করেই হোক, অন্ততপক্ষে কাজ চালানোর মত ভাষা আপনাকে শিখতেই হবে। আপনি এক দিনে কিন্তু জার্মান ভাষা আয়ত্ত করতে পারবেন না। যত দিন যাবে, আপনি যদি অনুশীলন করতে থাকেন, তাহলে নতুন নতুন শব্দ, ভাষার ব্যাকরণ সব আয়ত্ত হতে থাকবে। তবে আপনি কোনো দিনও বলতে পারবেন না যে আমি ১০০ ভাগ জার্মান জানি বা আমি ১০০ ভাগ স্প্যানিশ জানি। আজ পর্যন্ত বাংলা ভাষার ক্ষেত্রেই এমনটা বলার সাহস আমার হল না। প্রোগ্রামিং ল্যাংগুয়েজও একই রকম। আপনি যত দিন যাবে তত দক্ষ হতে থাকবেন, নতুন নতুন ট্রিক্স শিখবেন, কিন্তু কখনোই বলতে পারবেন না যে, আমি সব শিখে ফেলেছি।

প্রোগ্রামিং ল্যাংগুয়েজের সব শেখার দরকার ও কখনো হবে না। কারণ আপনাকে কথা বলতে হবে মেশিনের সাথে, মানুষের সাথে না। আর মেশিন মানুষের থেকে অনেক কম কমপ্লিকেটেড। এরা আপনাকে সহজ ভাবে কথা বলার জন্য তাচিল্য করবে না। এখানে একটা কথার একটাই অর্থ, আবার অনেক গুলো কথার একটা অর্থ হতে পারে। তবে মানুষের ভাষার মত একটা কথার অনেক গুলো অর্থ কখনোই হবে না। এখানে আপনি ভাষা পাবেন ঠিকই, কিন্তু সেই ভাষার কোন এক্সপ্রেশন নেই। যেটা বলবেন, সেটার অর্থ শুধু সেটাই হবে। তাই আমি মনে করি মানুষের ভাষা শেখার থেকে কম্পিউটারের ভাষা শেখা অনেক সহজ।

যখন আপনি প্রথম প্রোগ্রামিং ল্যাংগুয়েজ শেখা শুরু করবেন তখন খুবই বিরক্ত লাগবে। প্রোগ্রামিং ল্যাংগুয়েজ জানলে পুরো দুনিয়া কাঁপিয়ে ফেলা যায়, নাসাকে হ্যাক করে ফেলা যায়, গেম সফটওয়্যার কত কি বানানো যায়, এই রকম অসংখ্য বড় বড় কথার ফুলবুরি থেকে প্রাপ্ত নির্যাশ মাঝে স্বপ্ন নিয়ে আপনি এসেছেন প্রোগ্রামিং করতে। আর আপনাকে ধরিয়ে দেওয়া হয়েছে একটা সাদা আর একটা কালো টেক্সট এডিটর। সাদা টেক্সট এডিটরটাতে বলা হয়েছে  $a + b$  এর যোগফল বের করতে আর কালো টেক্সট এডিটরে (টার্মিনাল) বলা হয়েছে রান করে দেখতে যে, কোডটি সঠিক ভাবে কাজ করে কিনা, সঠিক আউটপুট দেয় কিনা। এই সব দেখলে যে কারোরই বিরক্ত লাগবে, অন্ততপক্ষে আমার মত যারা নাসাকে হ্যাক করার উদ্দেশ্যে প্রোগ্রামিং শিখতে আসবে তাদের তো স্বপ্ন ভেঙ্গে চুরমার হয়ে যাবে। মনে হবে পুরো পৃথিবীটাই ধোঁকা।

আমি কিন্তু এখানে আমার ওই সময়ের মনের অবস্থাটায় প্রকাশ করার চেষ্টা করছি। আমি কোন ভাবেই মনে নিতে পারছিলাম না যে এটাই প্রোগ্রামিং, কালো স্ক্রিনে স্লাস থ্রি ফোরের কয়েকটা বেসিক ম্যাথমেটিক্স আজেবাজে কিছু সাংকেতিক চিহ্ন দিয়ে প্রকাশ করে আমার কি

---

এমন লাভ হবে, কিভাবে আমি একটা সফটওয়্যার এর সান্ধাজ গড়ে তুলতে পারি এই সহজ অংক গুলোকে কঠিন ভাবে সমাধান করে। কোনো ভাবেই যখন কিছু বুঝে আসছিল না তখন আমাকে সংগ দিতে চলে এলো হতাশা, আর এই হতাশা কোনো দিনও আপনাকে আপনার লক্ষ্যে পৌঁছতে দিবে না। আমার শুধু এই একটা প্রশ্নই ছিল যে, কি হবে এই কালো স্ক্রিনে ছোট ছোট এই সব প্রশ্নের এর সমাধান করে? গ্রাফিক্স ছাড়া কিভাবে একটা সফটওয়্যার বানানো যায়? কিভাবে গেম বানানো যায়? যার উত্তর আজকে আমার জানা, আসলে তখন যেটা করতাম সেটাই হচ্ছে প্রোগ্রামিং, আর যা করবো বলে স্বপ্ন দেখতাম তা হল ডেভেলপমেন্ট।

প্রথমে আমাদেরকে রাজ্য জয় করতে হবে, কম্পিউটারের ভাষাটা পুরোপুরি আয়ত্ত করে কম্পিউটার নামক বোকা মেশিনটাকে নিজেদের গোলাম বানাতে হবে, তারপরে আমরা সিংহাসনে বসে তাকে যা খুশি তাই নির্দেশ দিতে পারব। তখন আমরা গেম, সফটওয়্যার সবই বানাতে পারব। আর এই রকম ছোট ছোট অংক সমাধান করতে দেওয়া হয় এই জন্যই যে একবারে বড় কিছু আমাদের শেখালে কোন কিছুই আমাদের মাথায় চুকবে না, সব মাথার ওপর দিয়ে চলে যাবে। আবার কম্পিউটার যেহেতু একটা বোকা মেশিন, তাকে বোঝানোর জন্য আমাদের সরাসরি কথা বলতে হবে। ঘুরিয়ে পেঁচিয়ে কথা বললে সে বুঝবে না। সে শুধু বোঝে লজিক, আর লজিক মানেই হচ্ছে অংক। তার মানে এই ছোট ছোট যেই অংক গুলো আমাদের প্রথমে সমাধান করতে খুব বিরক্ত লাগে, সেই অংক গুলোই আমাদের সারাজীবন কাজে লাগবে। কম্পিউটার কথাটা এসেছে কম্পিউটিং থেকে। কম্পিউটিং মানে হচ্ছে গণনা করা। আর গণনা করার জন্য আমাদের শিখতে হয় গণিত। অন্যভাবে বললে, কম্পিউটার সাইন্স নিয়ে কাজ করতে হলেও আমাদের দরকার গণিত।

এইসব কথাতো আমি শুরু করার ৪-৫ বছর পরে এসে বুঝতে পেরেছি। যদি আগে বুঝতে পারতাম তাহলে হয়ত যাত্রাটা আর একটু সহজ হতো। আমি যখন হতাশার প্রেমে ডুবে গেছি, প্রোগ্রামিং আমার দ্বারা হবে না বলে বই পত্র সব গুচ্ছিয়ে তুলে রেখেছি, হঠাৎ তখনই মনের কোনো এক কোণা থেকে জিদ উঁকি দিয়ে বলল, হবে না মানে? হতেই হবে। অনেক হয়েছে হতাশার সাথে প্রেম। এবার জিদকে নিজের সঙ্গী করে নেবার পালা। নতুন সঙ্গী পেয়ে নিজের জীবনটাকে আর একটু গুচ্ছিয়ে নিলাম।

আমার একটা জটিল রোগ ছিল। আমার মনে হয় আপনাদের ভিতরে অনেকেই এই একই রোগের রোগী। কোন কিছু সম্পূর্ণ ভাবে না বুঝে আমি সামনে আগাতে পারতাম না। আর

---

প্রোগ্রামিং এ প্রতি পদে পদেই নতুন নতুন কনফিউশন এসে উঁকি মারে। নতুন নতুন এমন সব নামের আবির্ভাব হয় যা পড়তে গেলে মনে হয় দাঁত ভঙ্গে যাবে। সেই সমস্যাকে দূরে সরিয়ে রেখে পাশে জায়গা দিলাম নোটবুককে। ধরে নিলাম, একবারেই সব কিছু জানা যায় না। হয়ত আমি এখন যেটা পারছি না, সেটা সামনে ঠিকই জানতে পারব। যখন যেই সমস্যায় পড়তাম সেটা দুইটা জায়গায় লিখে রাখতাম। প্রথমে লিখতাম নিজের মনে, তারপরে লিখতাম নোটবুকে। কনফিউশনের সাগরে না ভসে এবার সিন্ড্রান্ট নিলাম, আমার সব কিছু জানতে হবে না। যেহেতু বইতে লেখা আছে যে এইভাবে প্রোগ্রামিং করতে হয়, তাই এভাবেই আমাকে আগাতে হবে। সফটওয়্যার কিভাবে বানাতে হয় আমার জানতে হবে না। আমাকে শুধু প্রোগ্রামিং শিখতে হবে। যেদিন থেকে জিদ আমার সঙ্গী হয়েছে, সেদিন থেকে আর কোনো কনফিউশন, আর কোনো হতাশা আমাকে ছুঁতে পারেনি। আমার তখন লক্ষ্য একটাই ছিল, আমাকে একজন প্রোগ্রামার হতে হবে।

আপনি যখন প্রোগ্রামিং শেখা শুরু করবেন, তখন আশেপাশে কারোর দিকে তাকাবেন না। মনে রাখবেন সবার প্রথমে কমিউনিকেশন স্কিল দরকার। আপনি যদি কম্পিউটারকে নিজের বন্ধু বানাতে পারেন, তার সাথে নিজের মনের কথা শেয়ার করতে পারেন, তাহলে আপনি কম্পিউটার ব্যবহার করে যেকোনো কিছুই করতে পারবেন, এটা শুধু সময়ের অপেক্ষা মাত্র। প্রোগ্রামিং শিখতে এসে হতাশার কবলে পড়া আপনিই প্রথম ব্যক্তি না। আপনার পূর্বে যুগে যুগে বহু মানুষ এই হতাশার কবলে পড়েছে। এমনকি আজকে যাদেরকে আপনি প্রোগ্রামিং এর গুরু মনে করেন, তারাও এক সময় এই হতাশার সাথে গভীর প্রণয়ে নিমজ্জিত ছিল। প্রোগ্রামিং শুরু করার পরে আপনাকে দুইটা প্রতিজ্ঞা করতে হবে। যতই হতাশা আপনাকে প্রেম নিবেদন করুক না কেন, কোনো ভাবেই আপনি তার প্রেমে পড়বেন না। আর সব কিছু একবারেই জানতে বা বুঝতে চাইবেন না। আপনার সমস্যা গুলো মাথায় লিখে রাখুন, কিছু দূর আগালেই আপনি তার সমাধান পাবেন। আর কিছু দিন পর পর পূর্বে কি শিখেছিলেন সেগুলো পিছনে তাকিয়ে দেখবেন। স্টিভ জবস এর ভাষ্য মতে ডট মেলাবেন।

কম্পিউটার সাইন্স একটা বিশাল সমৃদ্ধি। এই বিশাল সমৃদ্ধি যদি আপনি সঠিক ভাবে সার্ভাইভ করতে চান তাহলে, আপনার দরকার পড়বে ধৈর্য, অধ্যাবশায় এবং লেগে থাকার মানুষিকতা। এর সাথে সাথে আরও দরকার পরবে একটা সঠিক দিক নির্দেশনা। কারণ এত বিশাল সমৃদ্ধি দিক হারিয়ে তাল বেতাল হয়ে যাওয়ার সন্তানটাই সব থেকে বেশি থাকে। আর আমাদের দেশের প্রেক্ষাপটে এটা হরহামেশাই ঘটে থাকে। এর সাথে সাথে আপনার দরকার পড়বে ইংরেজির কিছু জ্ঞান, কারণ এই জগতের সাথে যুক্ত সমস্ত দলিল ইংরেজিতেই

---

লেখা। আর অন্ন কিছু গণিতের দক্ষতা। গণিতের দক্ষতা ঠিক কত হলে ভালো হয় এর নির্দিষ্ট কোনো মাপকার্তি নেই। আপনি নিজেই ধীরে ধীরে বুঝতে পারবেন যে গণিতে আপনার দক্ষতা বৃদ্ধি করা উচিত কি না? তবে শুরু করার জন্য নূন্যতম একটা জ্ঞান আপনার দরকার হবেই। কারণ আমাদের এই বোকা বাক্তা কিছু না বুঝলেও গণিত খুব ভালো বোঝে।

আমি এই বইতে আপনাদেরকে একটা সঠিক গাইডলাইন দেওয়ার চেষ্টা করবো। যেই গাইডলাইনটা মনে চললে আপনি নিজেই কারোর মুখাপেক্ষী না হয়েও কম্পিউটার জগতে নিজের একটা অবস্থান তৈরি করে নিতে পারবেন। এই গাইডলাইনটা মোটেও ছোট কোনো গাইডলাইন নয়, আবার সবাইকে যে সব কিছু মানতে হবে এমনটাও নয়। প্রথমেই আপনাকে সিদ্ধান্ত নিয়ে নিতে হবে যে আপনি আসলে কত দূর দৌড়াতে চান, নিজের লক্ষ্য অনুযায়ী আমার গাইডলাইনটি ফলো করবেন। এই গাইডলাইনে আমি, প্রোগ্রামিং জগতে পদার্পণ থেকে শুরু করে কিভাবে একজন ভাল ফুলস্ট্যাক ওয়াবে ডেভেলপার হওয়া যায় এবং তার সব থেকে নূন্যতম রাস্তা কোনটা সেই বিষয়েই আলোকপাত করবো। সাথে সাথে আমাদের দেশের প্রচলিত ডেভেলপমেন্ট রিলেটেড কিছু মিথ নিয়েও আলোচনা করবো। এমন কিছু বিষয়ে আলোচনা করবো যেগুলো সত্য কিন্তু আপনার শুনতে ভালো লাগবে না।

একজন ভালো সফটওয়্যার ইঞ্জিনিয়ারের যে যে বিষয়ে জ্ঞান অর্জন করতে হয়, এক্স্যাক্ট কি কি কাজ করলে আপনাকে একজন ভালো ওয়াবে ডেভেলপার বলা হবে, তার সব কিছুই যুক্তি সহকারে আপনাকে বোঝানোর চেষ্টা করবো। যতটুকু আপনার পছন্দ হবে ঠিক তত টুকুই আপনি গ্রহণ করবেন। আর যদি কোনো বিষয়ে দ্বিমত থাকে, মন্তব্য করার পূর্বে ইমোশনকে দূরে রেখে যুক্তি দিয়ে চিন্তা করবেন। তবে আবারও একটা কথা আপনাকে স্মরণ করিয়ে দেই, ভালো ভাবে কম্পিউটার চালানো শেখার পরেই কম্পিউটারকে নিজের গোলাম বানাতে চাওয়া উচিত। তার আগে কোনো ভাবেই নয়। ভালো খারাপ সব জায়গাতেই আছে, অনেক কম জ্ঞান অর্জন করেও অনেকে ভালো উপার্জন করে সুখেই দিন কাটাচ্ছে। কিভাবে আপনি সহজে অনলাইনে অর্থ উপার্জন করবেন সেই তথ্য বিতরণের জন্য এই বইটা না। এই বইটি নিজেকে বাংলাদেশের একজন গবীত সফটওয়্যার সৈনিক হিসেবে গড়ে তোলার অকাট্য দলিল। যা ব্যবহার করে আপনি নিজেই নিজেকে স্বশিক্ষায় শিক্ষিত করে একজন মানব সম্পদে রূপান্তরিত হতে পারবেন।



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>



অধ্যায় দুই

## বোকা বাঞ্ছের ভাষা

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ প্রোগ্রামিং ল্যাংগুয়েজ কি
- ✓ কিভাবে শুরু করা উচিত
- ✓ কম্পিউটিটিভ প্রোগ্রামিং এর গুরুত্ব কতটা
- ✓ সি প্রোগ্রামিং কিভাবে শেখা উচিত
- ✓ সি রোডম্যাপ
- ✓ জাভা কিভাবে শেখা উচিত
- ✓ জাভা রোডম্যাপ

---

কম্পিউটারের সাথে বন্ধুত্ব করার, রাত জেগে তার সাথে গল্ল করার এক মাত্র মাধ্যম যে প্রোগ্রামিং ল্যাংগুয়েজ, সেটা আমরা এর মধ্যেই জেনে গিয়েছি। আমাদের দেশের সরকারও ব্যাপারটাকে খুব ভাল ভাবেই বুঝতে পেরেছেন। তাই স্কুল কলেজের পাঠ্যসূচীতেও তিনি এই প্রোগ্রামিং ল্যাংগুয়েজকে যুক্ত করেছেন। ইউনিভার্সিটিতে ভর্তি হওয়ার সাথে সাথেই হাতে গুঁজিয়ে দেওয়া হয় প্রোগ্রামিং এর ওপরে কয়েকটা কোর্স। বড় ভাইদের কাছে যদি কোনো সাজেশন চাই, তাহলে তারাও শুনিয়ে দেয় প্রোগ্রামিং ছাড়া কোনো গতি নেই, প্রোগ্রামিং শিখতেই হবে। প্রোগ্রামিং, প্রোগ্রামিং, প্রোগ্রামিং। ইউনিভার্সিটির প্রথম ১-২ টা বছর চারপাশে শুধু এই একটা শব্দই কানে বাজতে থাকে। ঘুমালেও ঘুমের ভিতরে দুঃস্বপ্ন দেখি, কানের কাছে কে যেন ভয়ংকর শব্দ করে প্রোগ্রামিং প্রোগ্রামিং করে চিল্লাচ্ছে। যত দিন যেতে থাকে, তত ভয় বাড়তে থাকে। কিছুই তো শেখা হল না, কি হবে আমার? চারপাশে গুজব ছড়ানোর, ভয় দেখানোর মানুষেরওতো তেমন অভাব দেখি না। যে যার মত করে দার্শনিক সব বুলি আওড়াচ্ছে। আমি ও কম যায় না, কোনটা সঠিক কোনটা বেষ্টিক একটু গুগলে সার্চ করে দেখে নেওয়ার মত সময় আমার নেই। এত মানুষকি তাহলে ভুল বলছে?

অনেক সময় অনেক মানুষ বললেই কথাটা সত্যি হয়ে যায় না। আজকের দিনে আমাদের অনেক কিছুর অভাব থাকলেও অন্ততপক্ষে ইন্টারনেটের অভাব নেই। ঘরে বাইরে, বাসে ট্রাকে, রিক্সায়, হাঁটতে হাঁটতে এমনকি ফুটবল খেলতে খেলতেও আমরা ফেসবুক ব্যবহার করি। তারমানে অল্প একটু মেগাবাইট খরচ করে, গুগলে সার্চ করে কোনটা সঠিক আর কোনটা বেষ্টিক জেনে নেওয়ার মত সামর্থ্য এখন আমাদের বেশিরভাগ মানুষেরই আছে। এবং আমি আশা করবো এর পর থেকে স্বেচ্ছাতে গা না ভাসিয়ে সঠিক তথ্যটা আপনারা নিজেরাই খুঁজে বের করবেন। তবে এখানে আপনাদের জন্য আমি কিছু প্রোগ্রামিং বিষয়ক ইনফরমেশন শেয়ার করছি, পরে যখন নিজে সার্চ করে দেখবেন, তখন আমার ইনফরমেশন গুলো সঠিক না ভুল সেটা অবশ্যই জানাবেন।

প্রোগ্রামিং হচ্ছে কম্পিউটারের জন্য লেখা কিছু নির্দেশনা। যা মূলত কম্পিউটারের ভাষায়ই লেখা হয় এবং কম্পিউটার সেই নির্দেশনা গুলো বুঝে সেই অনুযায়ী কাজ করে থাকে। সহজ ভাষায় বললে, কম্পিউটারকে আমরা যে নির্দেশ দেই সেটাই হচ্ছে প্রোগ্রামিং। আমরা কম্পিউটারকে যে কোন নির্দেশ দিতে পারি, তবে নির্দেশ গুলো সঠিক ভাবে দিতে হবে। এমন ভাবে নির্দেশ দিতে হবে যেন কম্পিউটার সেটা বুঝতে পারে। আর কম্পিউটার শুধু বুঝতে পারে তার নিজের ভাষা। তাহলে এখন নিশ্চয়ই বুঝতে পারছেন, আমাদেরকে প্রোগ্রামিং শিখতে হবে না, শিখতে হবে প্রোগ্রামিং ল্যাংগুয়েজ। কিভাবে কমান্ড দিলে এই বোকা

মেশিনটা আমাদের কমান্ড বুঝতে পারবে সেই কমান্ড দেওয়ার ভাষাটাই শিখতে হবে। কমান্ড দেওয়াতে আমরা এমনিতেই ওস্তাদ।



### What is Programming?

- Computer programming is the process of designing and building an executable computer program to accomplish a specific computing result. ([Wikipedia](#))

এখানেই সবাই পরে যায় মহা বিপাকে, আর জন্ম নেয় গুজব তত্ত্বের। ভাষাতো শিখব কিন্তু শিখবো কোনটা? গুগলে সার্চ করলে তো অন্ততপক্ষে কয়েকশ ভাষার নাম পাওয়া যায়। ইউনিভার্সিটিতে বলে সি শিখতে, এক বড় ভাই বলে পাইথন শিখতে তো আর একজন বলে জাভা। কেউ আবার বলে এসব শিখে লাভ নেই, পিএইচপি শেখ। কেউ তো জাভাস্ক্রিপ্ট ছাড়া দুনিয়াতে কোনো ভাষাকে ভাষায় মনে করে না। কেউ কেউ তো আবার খুব অ্যাডভান্সড, বলে সব শেখ। কিন্তু আমি কোনটা শিখব? কোনটা শিখলে আমার ভবিষ্যৎ উজ্জ্বল হবে? কোনটা শিখলে দ্রুত ডেভেলপমেন্ট করতে পারবো?

যে কোনো একটা শিখুন, কোনো সমস্যা নেই। যে কোন একটা ল্যাংগুয়েজ বেছে নিয়ে অনেকটা সময় সেই ল্যাংগুয়েজ শেখার পিছনে ব্যয় করুন। আসলে কম্পিউটার কিন্তু সি বোঝে না, সে জাভা, জাভাস্ক্রিপ্ট, পাইথন কিছুই বোঝে না। সে শুধু বোঝে বাইনারি, মানে ০ এবং ১। কিন্তু এই ভাবে ০ এবং ১ দিয়ে যদি কম্পিউটারকে আমরা নির্দেশ দিতে চাই, তাহলে আমাদের নিজেদেরই ১২টা বেজ যাবে। তাই আমরা হাই লেভেল কিছু ল্যাংগুয়েজ শিখে সেগুলো ব্যবহার করেই কাজ চালিয়ে নেই। মাঝখানে কম্পাইলার নামক একটা বুদ্ধিমান মেশিন আছে, যে কিনা আমাদের হাই লেভেল প্রোগ্রামিং ও বুঝতে পারে আবার লো লেভেল মেশিন কোডও বুঝতে পারে। সেই আমাদের হয়ে কম্পিউটারের সাথে যোগাযোগ করে থাকে।



### What is Machine Code?

- Machine code is a computer program written in machine language instructions that can be executed directly by a computer's central processing unit (CPU). ([Wikipedia](#))



## What is Compiler?

- A compiler is a computer program that translates computer code written in one programming language (the source language) into another language (the target language). The name compiler is primarily used for programs that translate source code from a high-level programming language to a lower level language (e.g., assembly language, object code, or machine code) to create an executable program. ([Wikipedia](#))

আপনি যে কোনো একটা হাই লেভেল ল্যাংগুয়েজ শিখতে পারেন। পৃথিবীতে দুই ধরনের মানুষ আছে। এক ধরনের মানুষ প্রথমে কোন কিছু দ্রুত শুরু করতে চাই, পরে আস্তে আস্তে তার গভীরে প্রবেশ করতে চাই। আপনি যদি এই দলের একজন হয়ে থাকেন, তাহলে আমি বলবো আপনি পাইথন দিয়ে শুরু করতে পারেন। ছেট এবং সহজ একটা ল্যাংগুয়েজ, কিন্তু খুবই পাওয়ারফুল। এখনকার সময়ে এটা একটা হট ল্যাংগুয়েজ যা ব্যবহার করে মোটামোটি সব ধরনের কাজই করা যায়। দ্বিতীয় ধরনের মানুষেরা একটু ওল্ড ফ্যাশন টাইপের হয়, অনেকটা আমার মত। তারা প্রথম থেকেই সব কিছু বুঝে শুনে তারপরে সামনে আগাতে চাই। তাদের কোন কিছু করার এত তাড়া নেই। আস্তে আস্তে ধীরে সুস্থে সব কিছু বুঝেই সামনে এগিয়ে যেতে চাই। আপনি যদি এই দলের কেউ হয়ে থাকেন, তাহলে আমি বলবো আপনার সি অথবা জাভা দিয়ে শুরু করা উচিত। কারণ এই ল্যাংগুয়েজ গুলো তুলনা মূলক লো লেভেল। এই ল্যাংগুয়েজ গুলো দিয়ে শুরু করলে অনেক জটিল বিষয়ও প্রথম থেকেই আপনার নখদর্পনে চলে আসবে। তবে একটা জিনিস মাথায় রাখবেন, কোন ল্যাংগুয়েজ শেখার সময় ভাববেন না যে, এই ল্যাংগুয়েজ ব্যবহার করে আপনি অনেক অনেক ডেভেলপমেন্ট করবেন। তাহলে কিন্তু ভুল হবে। আপনি এখন ডেভেলপমেন্ট শিখছেন না, আপনি শিখছেন কিভাবে কম্পিউটারকে বসে আনা যায়, কিভাবে তার সাথে কথা বলা যায়, এর থেকে বেশি কিছুই না। ভাষা, ভাষার ব্যাকরণ শেখার পূর্বেই যদি আমরা সাহিত্য লেখার চিন্তা করি সেটা তো মনে হয় ঠিক হবে না।



## What is High Level Programming Language?

- In computer science, a high-level programming language is a programming language with strong abstraction from the details of the computer. ([Wikipedia](#))

আমরা অনেকেই প্রথম ল্যাংগুয়েজ শেখার সময় খুব বেশি কনফিউসড থাকি। এমন একটা ল্যাংগুয়েজ শিখতে চাই যেটা আজকে শিখে কালকেই ডেভেলপমেন্টের কাজ শুরু করা যাবে। কিন্তু এটা সম্পূর্ণ ভুল প্রসেস। বাঙালি হিসেবে জন্ম নিয়ে জন্মের পরই যদি আমাদের বাবা মা বাংলা না শিখিয়ে ইংরেজি শেখাতো তাহলে কেমন হতো বলেন তো? বেশির ভাগ জবের ক্ষেত্রেই তো আমাদের ইংরেজিই দরকার, তাই পেট থেকে পড়েই ইংরেজি শেখা শুরু করলে তো মন্দ হয় না। জবের ক্ষেত্রে কয়েক ধাপ এগিয়েই থাকলাম বরঞ্চ। তাই না? প্রথম ভাষাটা সবার জন্যই খুব গুরুত্বপূর্ণ। হোক সেটা মানুষের ভাষা, হোক সেটা কম্পিউটারের ভাষা। প্রথম ভাষা আপনি আপনার কর্ম ক্ষেত্রে ব্যবহার করছেন কিনা সেটা গুরুত্বপূর্ণ না। গুরুত্বপূর্ণ হচ্ছে এমন একটা ভাষা বেছে নেওয়া যা শিখলে প্রোগ্রামিং, প্রোগ্রামিং ফান্ডামেন্টালস, প্রোগ্রামিং ল্যাংগুয়েজ এবং এর ব্যবহার সম্পর্কে মোটামাটি ভালো একটা ধারণা তৈরি হয়। প্রথম ভাষা শেখার পিছনে ব্যয় করা সময়টাকে অপচয় মনে করবেন না, এটা হচ্ছে আপনার টেকনোলজি জগতে নিজের অবস্থান পাকাপোক্ত করার প্রথম ইনভেষ্টিমেন্ট।

জীবনে আপনার অনেক প্রোগ্রামিং ল্যাংগুয়েজ শেখার প্রয়োজন পরবে। একটা ল্যাংগুয়েজ শিখেই আপনি আপনার জীবন পার করতে পারবেন না। তবে পূর্বেই বলেছিলাম, কম্পিউটারের ভাষা মানুষের ভাষার থেকে অনেক সহজ। তাই প্রোগ্রামিং ল্যাংগুয়েজের ফান্ডামেন্টাল বিষয় গুলো একবার আয়ত্ত করতে পারলেই আপনি যখন খুশি তখন যেকোনো ল্যাংগুয়েজ শিখে নিতে পারবেন খুব অল্প সময়ের মধ্যেই। পৃথিবীতে হাজার হাজার প্রোগ্রামিং ল্যাংগুয়েজ আছে, তার ভিতরে অল্প কিছু ল্যাংগুয়েজই জনপ্রিয়তার শীর্ষে উঠেছে। তবে প্রতিটা ল্যাংগুয়েজেই নিজস্ব সত্ত্ব আছে, নিজস্ব ব্যবহার আছে। তা যদি না থাকতো তাহলে কোটি কোটি টাকা খরচ করে নতুন নতুন ল্যাংগুয়েজ মানুষ বানাতো না। একটা ল্যাংগুয়েজ দিয়েই কাজ চালিয়ে নিত। পৃথিবীতে কোনো ল্যাংগুয়েজই স্বয়ংসম্পূর্ণ না।

### যে বিষয় গুলো মনে রাখতে হবে

- ❖ প্রথম ল্যাংগুয়েজ বুঝে শুনে সিলেক্ট করতে হবে
- ❖ ল্যাংগুয়েজ শেখার সময় ডেভেলপমেন্ট নিয়ে চিন্তা করা যাবে না
- ❖ প্রথম ল্যাংগুয়েজ শেখার জন্য সময় ব্যয় করতে হবে
- ❖ সহজে শুরু করতে চাইলে পাইথন দিয়ে শুরু করা যেতে পারে
- ❖ কম্পিউটারকে বুঝতে চাইলে সি প্রোগ্রামিং দিয়ে শুরু করতে হবে

সি প্রোগ্রামিং হচ্ছে সকল প্রোগ্রামিং ল্যাংগুয়েজ এর মা, কারণ মডার্ণ সমস্ত প্রোগ্রামিং ল্যাংগুয়েজের জন্ম হয়েছে সি প্রোগ্রামিং থেকেই। তাই আমি মনে করি, সি ল্যাংগুয়েজের প্রতিষ্ঠাতা ডেনিস রিচি এর প্রতি সম্মান প্রদর্শন করে আমাদের সি প্রোগ্রামিং ল্যাংগুয়েজ দিয়েই শুরু করা উচিত। এই ল্যাংগুয়েজটা খুবই ছোট একটা ল্যাংগুয়েজ। এবং এটা শিখলে কম্পিউটার সাইন্সের অনেক জটিল বিষয় পরিষ্কার হওয়ার সাথে সাথে প্রোগ্রামিং ফান্ডামেন্টালস্টাও আপনার নথদর্পনে চলে আসবে। একটা সমস্যাকে কিভাবে একজন কম্পিউটার সাইন্টিস্টের দৃষ্টিতে দেখতে হয় সেটা আপনাকে শেখাবে এই ছোট ল্যাঙ্গুয়েজটা।

এখানে প্রথমেই আমাদের শিখতে হবে কিভাবে সি প্রোগ্রামিং এর জন্য ওয়ার্কিং এনভাইরমেন্ট তৈরি করতে হয়। কিভাবে কম্পাইলার সেটআপ করতে হয়? কিভাবে টেক্সট এডিটর ইন্সটল করতে হয়?

## জনপ্রিয় কিছু সি কম্পাইলার



আমি যখন প্রথম সি প্রোগ্রামিং শিখব বলে মনস্থির করি তখন সুবিন ভাই এর ব্লগ পরে শুরু করছিলাম। তখনও তার বই বাজারে পাবলিশ হয়নি। সেখানে একটা হ্যালো ওয়ার্ল্ড প্রোগ্রাম দিয়ে শুরু করেছিলেন তিনি এবং বলেছিলেন যদি এই প্রোগ্রামটা কেউ রান করতে না পারে তাহলে তার জন্য প্রোগ্রামিং না। আমি টানা দুই সপ্তাহ দেখে দেখে টাইপ করে, কপি পেস্ট করেও প্রোগ্রামটি রান করতে পেরেছিলাম না। পরে বুঝেছিলাম, আমার কম্পাইলার সঠিক ভাবে সেটআপ হয়েছিল না, আমি এনভাইরমেন্ট ভ্যারিয়েবল সেটআপ করেছিলাম না।

তাই মনোযোগ সহকারে প্রতিটা স্টেপ ভালোভাবে বুঝতে হবে। কোন কাজটা কেন করছেন গুগল করে ভালোভাবে বুঝে নিতে হবে। এবং প্রথমেই নিজের কাজ করার জন্য সুন্দর একটা এনভাইরমেন্ট তৈরি করে নিতে হবে। এটা আসলে আপনার কাজ করার ঘরের মত। আপনি ঘরটাকে যত সুন্দর করে সাজাবেন, সব কিছু যত পরিষ্কার পরিচ্ছন্ন থাকবে, যত বেশি গ্যাজেট থাকবে ততই কাজ করে আপনি মজা পাবেন। একই ভাবে কোড করার ইনভাইরমেন্টটা যত

---

সুন্দর করে সাজাবেন, যত ভালো একটা টেক্সট এডিটর ব্যবহার করবেন, যত সাপোর্টিভ ইউটিলিটি টুলস ব্যবহার করবেন ততই কোড করে মজা পাবেন। তবে প্রথমেই বড় সড় একটা IDE (Integrated Development Environment) ব্যবহার করাটা বুদ্ধিমানের কাজ হবে না। তাহলে পুরো সিস্টেমটা কিভাবে কাজ করছে সেটা আপনার অজানাই থেকে যাবে।

এর পরবর্তীতে আসবে ল্যাংগুয়েজ শেখার পালা। আপনাকে প্রথমেই ল্যাংগুয়েজ এর জাহাজ হয়ে যেতে হবে না। সময় নিয়ে ধীরে ধীরে প্রোগ্রামিং ফান্ডামেন্টালস গুলো আয়ত্ত করতে হবে। ভৃট করে বসে কয়েক ঘণ্টার মধ্যেই এগুলো শেষ করা যায় না। আপনি যদি পৃথিবীর সব থেকে বুদ্ধিমান মানুষও হয়ে থাকেন এবং কয়েক ঘণ্টার টিউটোরিয়াল দেখেই দাবি করে বসেন যে আপনি প্রোগ্রামিং এর ফান্ডামেন্টালস বুঝে গেছেন এবং সারা পৃথিবী যদি আপনার কথায় সমর্থন দেয়, তারপরেও আমি মানবো না। প্রোগ্রামিং ফান্ডামেন্টালস গুলো হচ্ছে আপনার ভিত্তি, আপনি যতটা সময় নিয়ে এই ভিত্তি প্রস্তর স্থাপন করবেন ততটাই মজবুত হবে আপনার ক্যারিয়ার। তাই তাড়াভুড়া না করে, সময় নিয়ে ফান্ডামেন্টাল গুলো শিখুন এবং চর্চা করুন। যখন মনে হবে আপনি সবকিছু ভাল ভাবেই বুঝে গিয়েছেন, তারপরেও অন্তত পক্ষে ৩ মাস এই গুলোই চর্চা করুন। বারবার একই কাজ করলে ভুলে যাবার প্রবণতাটা হ্রাস পায়। আর এই ফান্ডামেন্টাল নলেজ গুলো আপনি সারা জীবন কাজে লাগাতে যাচ্ছেন।

একটা বিষয় সব সময় মাথায় রাখবেন, একটা ১ তলা বিউল্ডিং তৈরি করতেও যেমন ভিত্তি স্থাপন করতে হয় তেমনি ২০ তলা বিউল্ডিং তৈরিতেও ভিত্তি স্থাপনের দরকার হয়। ১ তলা বিউল্ডিং এর ভিত্তি ২-৩ দিনের ভিতরেই স্থাপন করা সম্পূর্ণ হয়ে যায়। কিন্তু ২০ তলা বিউল্ডিং এর ভিত্তি স্থাপন করতে অনেক সময় লাগে। কিছু কিছু ক্ষেত্রে কয়েক মাসও সময় লেগে যায়। এই ভিত্তি স্থাপনের সময় বাইরে থেকে দেখে বোঝাব কোনো উপায়ই থাকে না যে এটা ২০ তলা বিউল্ডিং এর ভিত্তি স্থাপন হচ্ছে। এই সময় আপনার চারপাশে দাঁড়িয়ে থাকা একতলা বিউল্ডিং গুলো আপনাকে দেখে মুচকি মুচকি হাসতে পারে, আপনাকে নিয়ে তামাশা করতে পারে। কারণ তারা ৪-৫ দিনের ভিতরেই মাথা উঁচু করে দাঁড়িয়ে গিয়েছে। আর আপনার দৃশ্যমান কোনো কাজ এখনো হয় নি যা আপনি দুনিয়াকে দেখাতে পারেন। এই একতলা বিউল্ডিং গুলো জানে না যে তারা সর্বোচ্চ একটা বা দুইটা পরিবারেরই থাকার ব্যবস্থা করতে পারবে। আর আপনি যখন মাথা উঁচু করে দাঁড়াবেন তখন কম করে হলেও একশ পরিবার আপনার বুকে নিশ্চিন্তে তাদের রাত্রিযাপন করবে।

**বিঃদ্রঃ** ভিত্তি স্থাপনের সময় একতলা বিউল্ডিং এর সংস্পর্শ থেকে দূরে থাকবেন। আর আপনি মাথা উঁচু করে দাঁড়ানোর পর তারা নিজেরাই আপনার কাছাকাছি আসবে না।

---

একজন নব্য প্রোগ্রামার হিসেবে খুব বেশি বিষয় প্রথমেই জানার দরকার আছে বলে আমি মনে করি না। আমি মনে করি নিচের বিষয় গুলো শিখেই অন্ততপক্ষে ৩-৪ মাস অনুশীলন করা উচিত নতুন কোনো বিষয় শেখার পূর্বে -

**ডাটা টাইপস এবং ভ্যারিয়েবল:** প্রোগ্রামিং এর সব থেকে গুরুত্বপূর্ণ দুইটি বিষয় হচ্ছে ডাটা টাইপস (Data Types) এবং ভ্যারিয়েবল (Variable)। কম্পিউটারে আমরা সাধারণত ডাটা নিয়ে কাজ করি। আমাদের কম্পিউটার ঠিক কোন ধরনের ডাটা নিয়ে কাজ করতে পারে সেটা আমাদের প্রথমেই ভালোভাবে জেনে নেওয়া দরকার। এর সাথে সাথে কম্পিউটার কিভাবে মেমরিতে ডাটা স্টোর করে রাখে, বিট বাইটের হিসেব কিভাবে হয়, কিভাবে একটা নাম্বার বা ক্যারেক্টারকে (Character) কম্পিউটার তার ভাষায় স্টোর করে রাখে এই বিষয় গুলোও জানা জরুরি। প্রোগ্রামিং ল্যাংগুয়েজে ভিন্ন ভিন্ন যে ডাটা টাইপ আছে, সে গুলো কিভাবে কাজ করে বা সেগুলোর সাইজ ভিন্ন কেন এই রকম সমস্ত প্রশ্নের উত্তর প্রথমেই আমাদের জেনে নিতে হবে। এগুলো জানতে গিয়ে আপনারা একটা মজার বিষয় খেয়াল করবেন, আর সেটা হল ছোট বেলায় যেই গণিতকে জীবনে কোন কাজে লাগবে না বলে মনোযোগ সহকারে ফাঁকি দিয়ে পার করেছিলেন, আজ সেই গণিতের ভূমিকা কতটা আপনার জীবনে।

**অপারেটরসঃ** আমরা ম্যাথেম্যাটিকস (Mathematics) এ বিভিন্ন সাংকেতিক চিহ্ন যেমন যোগ বিয়োগ গুণ ভাগ ব্যবহার করে বিভিন্ন সমস্যার সমাধান করতাম। ঠিক তেমনি প্রোগ্রামিং ল্যাংগুয়েজেও অসংখ্য সাংকেতিক চিহ্ন রয়েছে বিভিন্ন সমস্যার সমাধান করার জন্য। এই সাংকেতিক চিহ্ন গুলোকে অপারেটর (Operator) বলে। আপনি এখন যেই প্রোগ্রামিং ল্যাংগুয়েজটা শিখছেন, সেই ল্যাংগুয়েজ আপনাকে কি কি অপারেটর প্রোভাইড করছে, কোন অপারেটরের কাজ কি, সবার প্রথমেই সেটি জেনে নেওয়া জরুরি। তাহলে আপনার অনেক কাজ সহজ হয়ে যাবে।

**কন্ডিশন বা ডিসিশন মেকিংঃ** আমরা জানি আমাদের এই বোকা বাক্তা খুবই লজিক্যাল। তাই তাকে ইন্সট্রাক্ট করতে গেলে আমাদেরকেও লজিক্যাল হতে হবে এবং জানতে হবে কিভাবে আমরা লজিক্যাল ভাবে তাকে ইন্সট্রাক্ট করতে পারি। বিভিন্ন ল্যাংগুয়েজে বিভিন্ন ভাবে এই লজিক্যাল অপারেশন গুলো লেখা যায়। তবে সব ল্যাংগুয়েজ এই if, if else, else if এবং switch স্টেটমেন্ট গুলো আপনি দেখতে পারবেন। চট করে এর ব্যবহার গুলো শিখে

---

নিবেন। কিন্তু প্রথম দিকে একটু কষ্ট হবে লজিক গুলো খুঁজে বের করতে, যদি তাই হয় তাহলে কিছু দিন সাধারণ ভাবে কোন কিছু চিন্তা না করে লজিক্যাল ভাবে চিন্তা করার প্রাকটিস করতে পারেন। যেমন আপনার কফি খাওয়ার ইচ্ছে হলে এভাবে বলতে পারেন যে, আমার মাথা ব্যাথা করলে আমি কফি খাব, না হলে প্রোগ্রামিং করবো। অফিস থাকলে সকাল ৭ টায় ঘুম থেকে উঠবো না থাকলে ১০টা পর্যন্ত ঘুমাবো। এভাবে যখন প্রতিটা বিষয় লজিক্যাল ভাবে চিন্তা করবেন আর চোখের সামনে ভেসে ওঠা লজিক্যাল কোড গুলো দেখবেন, আমার মনে হয় না খুব বেশি সময় লাগবে বোকা বাক্তকে লজিক শেখাতে।

**লুপ এবং রিপিটিশন:** আমি যখন প্রথম প্রোগ্রামিং ল্যাংগুয়েজ শিখি তখন সব থেকে বেশি সমস্যার সম্মুখীন হয়েছিলাম এই লুপ (Loop) বুঝতে গিয়ে। কোনো ভাবেই এর কনসেপ্ট মাথায় চুকচ্ছিল না। এর সিনট্যাক্স মনে রাখাটা খুব বেশি কঠিন কিছু ছিল না। কেন এবং কোথায় আমি লুপ ব্যবহার করবো সেটায় কোনো ভাবে বুঝতে পারছিলাম না। আসলে আমরা অনেকেই মনে করি প্রোগ্রামিং শেখার অর্থ হচ্ছে সিনট্যাক্স শেখা। কিন্তু না, সিনট্যাক্স আপনি মুখ্যত করেও কাজ করতে পারবেন। কয়েকদিন কোড লিখলে সমস্ত সিনট্যাক্স আপনার এমনিতেই মুখ্যত হয়ে যাবে। কিন্তু এই সিনট্যাক্স কোথায় এবং কেন ব্যবহার করবেন, এই সিনট্যাক্স আসলে কোন সমস্যার সমাধান দিচ্ছে এটা বুঝতে পারাটাই প্রোগ্রামিং। এটা শিখতেই সব থেকে বেশি সময় লাগে।

একই ধরনের কাজ বার বার করার জন্য ব্যবহার করা হয় লুপ, যেমন ১-১০০ পর্যন্ত স্ক্রিনে প্রিন্ট করা, আপনার নামটা ১০০০ বার স্ক্রিনে প্রিন্ট করা। যদি লুপ নিয়ে আপনিও প্রশ্নে ফেস করে থাকেন, তাহলে বলবো ধাবড়ে যাওয়ার কিছু নেই, এটা স্বাভাবিক। অন্ন কিছু প্রশ্নে সল্লু করলেই লুপ একদম পানির মত পরিষ্কার হয়ে যাবে। সব প্রোগ্রামিং ল্যাংগুয়েজেই মূলত তিনি ধরনের লুপ থাকে - for loop, while loop এবং do while loop। তবে সব থেকে বেশি ব্যবহৃত হয় for এবং while লুপ।

**ফাংশন:** ছোটবেলায় যখন গণিত বইতে ফাংশনের (Function) অধ্যায়টা আসত কোনভাবে ফাঁকি দিয়ে পার করতে পারলেই বাঁচতাম। আমি নিজেই অনেক বার বলেছি যে, এই ফাংশন আমার জীবনে কি কাজে লাগবে? তখন কি আর জানতাম যে আমি প্রোগ্রামার হব? ফাংশন অনেকটা লুপের মতই কাজ করে। একই ধরনের কোড বার বার না লিখে একবার লিখেই যতবার খুশি ততবার ব্যবহার করা যায় ফাংশনের মাধ্যমে। ফাংশন আর লুপের ভিতরে সব থেকে বড় পার্থক্য হচ্ছে, লুপকে আমরা কন্ট্রোল করতে পারি না, কিন্তু

---

ফাংশনকে যখন খুশি তখন রান বা এক্সিকিউট করাতে পারি। ফাংশন প্রোগ্রামিং এর একটা গুরুত্বপূর্ণ বিষয়। তাই ফাংশন এবং এর সাথে জড়িত বিষয়গুলো সম্পর্কে প্রথমেই একটা ভাল ধারণা তৈরি করে নিতে হবে।

**অ্যারে:** প্রোগ্রামিং এ আমরা সব সময় ডাটা নিয়ে কাজ করে থাকি। তাই অনেক সময় আমাদের একই রকম অসংখ্য ডাটা নিয়ে কাজ করার দরকার হয়। এই একই রকম অসংখ্য ডাটা একসাথে রাখার জন্য আমাদের একটা স্টোরেজ দরকার আর প্রোগ্রামিং এ সেই স্টোরেজটা হচ্ছে অ্যারে (Array)। প্রতিটা প্রোগ্রামিং ল্যাংগুয়েজে এই অ্যারে বা অ্যারের মত ডাটা স্ট্রাকচার রয়েছে। এটা হচ্ছে সব থেকে বেসিক একটা ডাটা স্ট্রাকচার। ছোটোখাটো কোনো সমস্যাও সমাধান করতে গেলে আমাদের অ্যারে দরকার পরে। তাই শুরুতেই অ্যারে এবং এর অপারেশন গুলো সম্পর্কে একটা ভালো ধারণা থাকা খুব বেশি জরুরি।

**স্ট্রিং:** কম্পিউটার সাধারণত সব কিছুই করে ম্যাথমেটিক্যাল দিয়ে, কিন্তু আমরা সাধারণ মানুষেরা অত ভাল ম্যাথমেটিক্যাল বুঝি না। আমাদের জন্য ভাল আমাদের নিজেদের ভাষা। কিন্তু কম্পিউটারতো আবার আমাদের ভাষা বুঝবে না। এই জন্য বিভিন্ন ক্যারেক্টোর ইনকোডিং ব্যবহার করে আমাদের ভাষা কম্পিউটারকে বোঝানো হয়। একজন প্রোগ্রামারের দায়িত্ব মানুষের সমস্যা গুলোকে কম্পিউটারের ব্যবহার করে সমাধান করা, আর সেই কাজ করতে হলে আমাদেরকে ভালভাবে স্ট্রিং (String) নিয়ে কাজ করা বুরুতে হবে। স্ট্রিং হচ্ছে অনেক গুলো ক্যারেক্টোরের সমষ্টি বা অ্যারে। এখানে আমি যা লিখছি সব কিছুই স্ট্রিং আকারে কম্পিউটারের কাছে স্টোর হচ্ছে। একটা স্ট্রিং কিভাবে কম্পিউটার তার ভাষায় কনভার্ট করে, কিভাবে স্ট্রিং অপারেশন গুলো ঘটে থাকে, কিভাবে দুইটা স্ট্রিং জোড়া লাগিয়ে একটা বড় স্ট্রিং তৈরি করতে হয়, কিভাবে একটা স্ট্রিং থেকে কিছু ওয়ার্ড খুঁজে বের করে আনতে হয়, এইরকম ছোটোখাটো কাজ আমাদের শিখতে হবে।

**পয়েন্টার:** কম্পিউটার সাইন্সের খুবই গুরুত্বপূর্ণ একটা টপিক্যাল হচ্ছে এই পয়েন্টার (Pointer)। প্রতিটা ল্যাংগুয়েজেই ডিরেক্টলি অথবা ইন্ডিরেক্টলি এই কনসেপ্টটা আমরা পেয়ে থাকি। এর আর এক নাম হচ্ছে রেফারেন্স। আমাদের ডিক্লিয়ার করা ভ্যারিয়েবল গুলো র্যামের কোন লোকেশনে রাখা আছে তার রেফারেন্স ধারণ করে রাখার কাজ করে এই পয়েন্টার। এটা একটু অ্যাডভান্সড টপিক্যাল, কিন্তু প্রথম থেকেই যদি শেখা শুরু করেন তাহলে আয়ত্ত হতে খুব বেশি সময় লাগার কথা না। কিভাবে পয়েন্টার কাজ করে, কিভাবে ডিক্লিয়ার করতে হয়, কিভাবে বিভিন্ন অপারেশন ঘটাতে হয়, পয়েন্টার আর অ্যারে এর মধ্যে সম্পর্ক কি,

---

পয়েন্টার ব্যবহার করে যেকোনো জায়গা থেকে কিভাবে একটা ভ্যারিয়েবলের ভ্যালু পরিবর্তন করা যায় এই রকম অসংখ্য কাজ আছে যা আমাদের শিখতে হবে।

**মেমরি অ্যালোকেশন:** হাই লেভেল প্রোগ্রামিং ল্যাংগুয়েজ গুলোতে মেমরি অ্যালোকেট (Memory Allocation) করা বা গার্বেজ মেমরি (Garbage Collection) ক্লিয়ার করা নিয়ে আমাদের চিন্তা করতে হয় না। তাই কিছু শিক্ষা সব সময়ের জন্যই অধরা থেকে যায়। কিন্তু সি, সি++ এর মত মিড লেভেল ল্যাংগুয়েজে আমাদের নিজে থেকেই মেমরি অ্যালোকেট করতে হয়। যেই মেমরি আর কাজে লাগছে না সেটাকে ফ্রি করতে হয়। এটা খুব কঠিন কোনো কাজ না। প্রয়োজন অনুযায়ী আমরা র্যাম এ নতুন মেমরি বরাদ্দ করে দিতে পারি মেমরি অ্যালোকেশন এর মাধ্যমে। প্রথম দিকে এটা খুব কঠিন মনে হবে, কারণ আপনি এর আসল ব্যবহার খুঁজে পাবেন না। আসলে ছোটোখাটো কাজের জন্য এগুলোর প্রয়োজন হয় না। বড় অ্যাপ্লিকেশন যখন তৈরি করবেন তখন এরকম অ্যাডভান্সড ফিচার গুলো দরকার পরবে। তারপরেও শিখে রাখলে হাই লেভেল ল্যাংগুয়েজ গুলোর কাজ বুঝতে অনেক সহজ হবে।

**স্ট্রাকচার এবং ইউনিয়ন:** যখন একই ধরনের অনেক গুলো ডাটা একসাথে রাখার দরকার হয় তখন আমরা অ্যারে ব্যবহার করে থাকি। কিন্তু যখন ভিৱ ভিৱ টাইপের অনেক ডাটা একসাথে রাখতে হয় তখন আমাদের দরকার হয় স্ট্রাকচার (Structure) বা ইউনিয়ন (Union) এর। অনেক হাই লেভেল ল্যাংগুয়েজ এ যাকে আমরা অবজেক্ট (Object) বলে থাকি, সি ল্যাংগুয়েজ এ সেই কাজটাই স্ট্রাকচার বা ইউনিয়ন ব্যবহার করে করা হয়। ওপরের অন্যান্য ফান্ডামেন্টাল ফিচার গুলোর মত এটাও খুবই গুরুত্বপূর্ণ একটা টপিক্স। কারণ প্রোগ্রামিং ল্যাংগুয়েজে আমাদের সব সময় অসংখ্য ভিৱ ডাটা নিয়েই কাজ করতে হয়।

ওপরে বর্ণিত বিষয় গুলোই হচ্ছে প্রোগ্রামিং এর ফান্ডামেন্টাল বিষয়। আপনি দুনিয়ার যে কোন সমস্যার সমাধান করতে পারবেন এই ফান্ডামেন্টাল বিষয় গুলো ব্যবহার করে। তাহলে এত অ্যাডভান্সড ফিচার গুলো কেন এসেছে? এসেছে আপনার কষ্ট কমানোর জন্য। যদি আপনি শুধু ফান্ডামেন্টাল বিষয় গুলো ব্যবহার করে সব রকম সমস্যার সমাধান করতে চান, সেই ক্ষেত্রে একটা সমস্যা নিয়ে ভাবতে, তার সমাধান বের করতে এবং সেটাকে কোডে রূপান্তর করতে অনেক বেশি সময় লাগবে। কিন্তু আপনি যদি কেবল প্রোগ্রামিং ল্যাংগুয়েজ শেখা শুরু করেন তাহলে আপনার একমাত্র কাজই হবে প্রোগ্রামিং এর ফান্ডামেন্টাল বিষয় গুলোকে আয়ত্ত করা। টিউটোরিয়াল দেখে এই বিষয় গুলো শিখতে খুব একটা সময় লাগবে না।

সময় লাগবে এর ব্যবহার বুঝতে। আর প্রোগ্রামিং ল্যাংগুয়েজ, প্রোগ্রামিং ফান্ডামেন্টালস শিখতে বলা হচ্ছে এর অর্থ কোনো ভাবেই সিনট্যাক্স শিখতে বলা হচ্ছে না, বলা হচ্ছে এগুলোর ব্যবহার করা শিখতে। আপনি শুধুমাত্র তখনই প্রতিটা বিষয়ের ব্যবহার ভালোভাবে জানবেন, বুঝবেন যখন আপনি প্রতিটা ফিচার, প্রতিটা টপিক্স প্রচুর পরিমাণে ব্যবহার করবেন।

## সি প্রোগ্রামিং এ যা যা শিখতে হবে

### **Step 1 - Environment Setup**

- Install GCC Compiler
- Setup Environment Variable
- Install Text Editor
- Install Necessary Plugins

### **Step 2 - Basic Syntax**

- Variables
- Data Types
- Operators
- Input and Output

### **Step 3 - Conditionals**

- If, If Else
- Switch Statement
- For Loop
- While, Do While Loop
- Break, Continue, Goto

### **Step 4 - Functions**

- Function Declaration
- Arguments and Parameter
- Return Value
- Function Scope

### **Step 5 - Arrays and String**

- Array Declaration
- Array Traversing
- Multi Dimensional Array
- String Declaration
- String Operations

### **Step 6 - Pointers**

- Pointer Declaration
- Pointer Operations
- Array and Pointer
- Memory Allocations
- Dynamic Array

### **Step 7 - Struct & Union**

- Struct Declaration
- Union Declaration

### **Step 8 - File Handling**

- Read, Write & Append
- Copy, Delete, Renaming

## কেন আমি মনে করি সি প্রথম ল্যাংগুয়েজ হওয়া উচিত?

- সি প্রোগ্রামিং ল্যাংগুয়েজটি হাই লেভেল ল্যাংগুয়েজ হলেও মেশিনের খুব কাছাকাছি একটা ল্যাংগুয়েজ। সাধারণত একে মিড লেভেল ল্যাংগুয়েজ বলা হয়। তাই এখানে কাজ করতে হলে মেশিনের অনেক কিছু সম্পর্কেই ভালো ধারণা থাকতে হয়। আর এই বিষয় গুলো সি ল্যাংগুয়েজ শিখতে শিখতেই শেখা হয়ে যায়।
- সি প্রোগ্রামিং ল্যাংগুয়েজে ভিন্ন ভিন্ন রকমের ডাটা টাইপ (Data Type) থাকলেও কম্পিউটার শুধুমাত্র ইন্টিজার টাইপের ডাটাই ধারণ করতে পারে। তাহলে ক্যারেক্টার (Char) বা ফ্লোট (Float) বা ইন্টিজার (int, long, short) টাইপের ডাটাই কম্পিউটার কিভাবে স্টোর করে? সি ল্যাংগুয়েজে ভিন্ন ভিন্ন ধরনের ইন্টিজার টাইপের ডাটা আছে সাইজের ওপরে ভিত্তি করে, কিন্তু কেন? কি ঘটে একটা ভ্যারিয়েবল ডিক্লেয়ার করলে? কিভাবে কম্পিউটার এত ডাটা স্টোর করে রাখে? এই বিষয় গুলো প্রতিটা ল্যাংগুয়েজে থাকলেও স্কিপ করে যাওয়ার সুযোগ সেখানে রয়েছে। কিন্তু সি প্রোগ্রামিং ল্যাংগুয়েজে আপনাকে এই বিষয় গুলো জানতে হবে।
- পয়েন্টার (Pointer) এবং অ্যারে (Array) একে অন্যের সাথে সম্পৃক্ত একটা বিষয়। অন্য ল্যাংগুয়েজ গুলোতে অ্যারে কিভাবে কাজ করে না বুঝেও আমরা সমস্ত কাজ করতে পারি। কিন্তু এখানে, কিভাবে অ্যারে কাজ করে, কিভাবে প্রতিটা এলিমেন্টের মেমরি অ্যালোকেট (Memory Allocation) হয়, কিভাবে আমরা অ্যালোকেটেড মেমরি থেকে ডাটা তুলে আনতে পারি, পরিবর্তন করতে পারি পয়েন্টার (রেফারেন্স) ব্যবহার করে সেগুলো সবই জানতে হয়। যার ফলে কম্পিউটার সাইন্সের সব থেকে গুরুত্বপূর্ণ একটা টপিক্স, আমাদের ব্যবহার করা প্রথম ডাটা স্ট্রাকচার সম্পর্কে একটা ভালো ধারণা তৈরি হয়ে যায়।
- আমরা যখন হাই লেভেল ল্যাংগুয়েজে কাজ করি, তখন স্ট্রিং (String) কে একটা ডাটা টাইপ ভেবেই কাজ করে যায়। বেশির ভাগ ল্যাংগুয়েজেই স্ট্রিং একটা ডাটা টাইপ। কিন্তু সি প্রোগ্রামিং ল্যাংগুয়েজ আমাদের ভাবতে বাধ্য করে যে

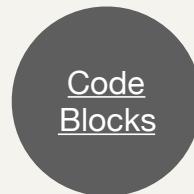
## কেন আমি মনে করি সি প্রথম ল্যাংগুয়েজ হওয়া উচিত?

এটা কোনো সিঙ্গেল ডাটা টাইপ না, এটা একটা ক্যারেক্ট্রার অ্যারে। সি প্রোগ্রামিং ল্যাংগুয়েজে স্ট্রিং নিয়ে কাজ করা তুলনা মূলক অনেক কঠিন। তবে এটা আমাদের শিক্ষাকে পরিপূর্ণতা দেই। যখন আমরা বাফার (Buffer) বা স্ট্রিম (Stream) নিয়ে কাজ করি তখন সি ল্যাংগুয়েজে শেখা এই টেকনিক গুলো অনেক কাজে লাগে।

- হাই লেভেল ল্যাংগুয়েজ গুলোতে যখন আমরা কাজ করি, খুব কম সময় আমাদের ক্যারেক্ট্রার ইনকোডিং (Character Encoding) নিয়ে জানতে হয়। কিন্তু সি ল্যাংগুয়েজে স্ট্রিং বা ক্যারেক্ট্রার নিয়ে কাজ করতে হলে আমাদের প্রথম থেকেই ক্যারেক্ট্রার ইনকোডিং সম্পর্কে ধারণা রাখতে হয়।
- সি প্রোগ্রামিং ল্যাংগুয়েজ খুবই ছোট্ট একটা ল্যাংগুয়েজ। খুবই ছোট্ট এর স্ট্যান্ডার্ড লাইব্রেরী (Standard Library)। তাই বেশির ভাগ প্রক্রিয়া এর সমাধান করতে হয় অন্ন কয়েকটা টুলস ব্যবহার করে নিজের বুদ্ধিমত্তাকে কাজে লাগিয়ে। যার ফলে প্রক্রিয়া সম্পর্কে জ্ঞানটা একেবারে সমৃদ্ধ হয়ে যায়।

**বিঃদ্রঃ** বিগিনার হিসেবে কখনোই কোড করার জন্য কোনো IDE (Integrated Development Environment) বেছে নেওয়া উচিত নয়। তাহলে একটা সিস্টেম কিভাবে কাজ করে তার বেশির ভাগ বিষয়ই আপনার অজ্ঞান থেকে যাবে। আপনি যেকোনো জনপ্রিয় কোড এডিটর ব্যবহার করে শুরু করতে পারেন।

### জনপ্রিয় কিছু কোড এডিটর এবং IDE



## সি প্রোগ্রামিং শেখার রেফারেন্স

### সি প্রোগ্রামিং এর বই সমূহঃ

- হাবলুদের জন্য প্রোগ্রামিং (ঝঁকার মাহবুব)
- কম্পিউটার প্রোগ্রামিং (তামিম শাহরিয়ার সুবিন)
- Beginning C by Ivor Horton (Appress)
- C: How to Program By Deitel, Deitel
- Programming in ANSI C By Balagurusamy
- The C Programming Language By Dennis Ritchie

### সি প্রোগ্রামিং এর ইউটিউব চ্যানেলঃ

- [Anisul Islam](#) (Bangla)
- [Tamim Shahriar](#) (Bangla)
- [Stack Learner](#) (Bangla)
- [Neso Academy](#) (English)
- [Caleb Curry](#) (English)

### সি প্রোগ্রামিং ওয়েবসাইটঃ

- [Geeks For Geeks](#)
- [C Programming](#)
- [Programmiz](#)
- [Learn-C](#)

### সি প্রোগ্রামিং ট্রেনিং প্রোগ্রাম এবং কোর্সঃ

- [Stack Learner Premium Courses](#)
- [Stack Learner Bootcamps \(Training Program\)](#)

বিঃদ্রঃ একটি নির্দিষ্ট সময়ে একটি নির্দিষ্ট রিসোর্স মনে চলবেন

---

বেশিরভাগ বিগিনারদের ক্ষেত্রে সব থেকে বড় সমস্যা হচ্ছে, প্রশ্নেম কোথায় পাব? প্রশ্নেম না পেলে সল্লু করবো কি? আর সল্লু না করলে ব্যাপার গুলো পরিষ্কার হবে কিভাবে? আর ব্যাপার গুলো পরিষ্কার না থাকলে সমস্যারই বা সমাধান করবো কিভাবে?

বিষয়টা একটু কনফিউসিং। প্রোগ্রামিং শুরু করার পরে এই একটা ধাপেই আপনি সব থেকে বেশি কনফিউসড হবেন, সব থেকে বেশি হতাশ হবেন। কিন্তু এই ধাপটা কোনভাবে পার করতে পারলেই আপনি এই জগতে নিজের নাম লেখানোর জন্য প্রস্তুত হয়ে যাবেন। এই সমস্যার সমাধান করতে হবে নিজের বুদ্ধিমত্তাকে কাজে লাগিয়ে। ঘণ্টার পর ঘণ্টার, দিনের পর দিন ছোট এই প্রোগ্রামিং স্কিল দিয়ে ছোট বেলায় মুখ্যস্ত করে আসা ম্যাথমেটিক্স এর জ্ঞান কাজে লাগিয়েই আমাদের প্রশ্নেম সল্লু করে যেতে হবে।

প্রশ্নেম সল্লুং এর কথা আসলে মানুষ দুই দলে বিভক্ত হয়ে যায়। একদল সরাসরি দাবি করে প্রশ্নেম সল্লুং এর কোন দরকার নেই, আর একদল এর দাবি প্রশ্নেম সল্লুং ছাড়া কোন ভাবেই সন্তুষ্ট নয়। আসলে আমার মনে হয়, এখানে আমাদের একটু বোঝার ভুল হচ্ছে। আপনি প্রোগ্রামিং করছেন মানে সারা জীবন আপনাকে প্রশ্নেম সল্লুই করে যেতে হবে। সেটা হতে পারে আমাদের বাস্তব জীবনের কোন প্রশ্নেম, আবার হতে পারে বিভিন্ন অনলাইন জাজে লিপিবদ্ধ করা কোন প্রশ্নেম। কিন্তু প্রশ্নেম আপনাকে সল্লু করতেই হবে। আমি মনে করি অন্তত পক্ষে ৫০০ প্রশ্নেম অনলাইন জাজে সল্লু করা উচিত। এতে করে আপনার প্রোগ্রামিং ফান্ডামেন্টালস এর যত কনসেপ্ট আছে, সব একেবারে পানির মত পরিষ্কার হয়ে যাবে। আর একটা বিষয়ও মাথায় রাখবেন। সেটা হচ্ছে অনলাইন জাজে প্রশ্নেম সল্লু করছেন মানেই আপনাকে কম্পিউটিভ প্রোগ্রামিং করতে হবে না। যদি আপনার কম্পিউটিভ প্রোগ্রামিং ভাল লাগে তাহলে তার জন্য প্রস্তুতি নিতে পারেন, আর না হলে নিজের বেসকে মজবুত করার জন্য অনলাইন জাজে দেওয়া প্রশ্নেম গুলো সমাধান করুন।

কম্পিউটিভ প্রোগ্রামিং মানে হল প্রোগ্রামিং ল্যাংগুয়েজ ব্যবহার করে প্রশ্নেম সল্লু করার একটা কম্পিউটিশন। আর এর একটা ভালো দিক হচ্ছে, অনেক দিন যাবত একটা ল্যাংগুয়েজ নিয়ে কাজ করার ফলে সেই ল্যাংগুয়েজ সম্পর্কে, প্রোগ্রামিং এর ফান্ডামেন্টালস সম্পর্কে খুব ভালো একটা জ্ঞান অর্জন হয়ে যায়। কারণ অনেকটা সময় সব রকম প্রশ্নেম সমাধান করতে হয় একটা নির্দিষ্ট ল্যাংগুয়েজ ব্যবহার করে। যার ফলে কর্ম জীবনে যে কোনো ল্যাংগুয়েজ নিয়ে কাজ করতে অথবা প্রয়োজন হলেই নতুন ল্যাংগুয়েজে শিফট করতে অনেক সহজ হয়।

---

এর আর একটা ভালো দিক হচ্ছে, আপনার সুন্দর একটা মানুষিকতা তৈরি হবে। যারা কখনো প্রশ্নে সল্ভিং করেন নি, তারা খেয়াল করলেই দেখবেন, ডেভেলপমেন্টে যখন আপনি একটা সমস্যার সম্মুখীন হন তখন অনেক ভয় লাগে। একটু ঘাবড়ে যান। সার্চ করে এর সমাধান খোঁজার চেষ্টা করেন। থার্ড পার্টি সল্যুশন, লাইব্রেরী খুঁজে বের করতে ব্যস্ত হয়ে পরেন। যদি থার্ড পার্টি কোনো সমাধান না পাওয়া যায় তখন খুব বেশি ফ্রাঞ্চেড লাগে। একজন কম্পিউটিভ প্রোগ্রামারের যেকোনো সময়, যেকোনো প্রশ্নে সমাধানের চেষ্টা করার একটা মানুষিকতা আগে থেকেই তৈরি হয়ে যায়। কারণ তারা জীবনের অনেকগুলো বছর ঘুমাতে যেত সমস্যা নিয়ে, আর ঘুম থেকে উঠত সমাধান নিয়ে। তাই কোনো প্রশ্নে তার কাছে প্রশ্নে না। থার্ড পার্টি সমাধান থাকুক আর না থাকুক, প্রয়োজন বোধে অ্যালগোরিদম ডেভেলপ করে হলেও একটা সমস্যার সমাধান করার মানুষিকতা রাখে একজন প্রশ্নে সল্ভিং সল্ভার। প্রশ্নে সল্ভিং করতে করতে এই অ্যাটিটিউডটা তাদের ভিতরে কখন যে ইনহেরিট হয়ে যায় তারা নিজেরাও জানে না। একটা সমস্যাকে সমাধান করার সব থেকে সহজ এবং কন্স্ট্রাকচিভ ওয়েটা খুঁজে বের করতে তারা ওস্তাদ হয়ে যায়।

আমরা অনেকেই মনে করি সমস্যার সমাধান হওয়াটাই সব কিছু। যখন একটা সমস্যার কোনো সমাধান নেই, তখন মুখ্য বিষয় এর সমাধান বের করা। কিন্তু যখন সমাধান হয়ে গেছে, তখন মুখ্য বিষয় এই কোডটার পার্ফরমেন্স অপ্টিমাইজ করা। কোডের সৌন্দর্য অনেক বড় একটা বিষয়। আপনার ভ্যারিয়েবল ডিক্লারেশন দেখেও একজন দক্ষ প্রোগ্রামার বলে দিতে পারবেন যে প্রোগ্রামিং এ আপনি কত দিন আছেন, আপনার দক্ষতা কেমন? কোডের সৌন্দর্য বলতে আসলে আপনি কিভাবে কোড লিখছেন, ফর্মেট করছেন বা ভ্যারিয়েবল ডিক্লেয়ার করছেন শুধু এটাই বোঝায় না। আপনি কোথায় ফাংশন ব্যবহার করছেন, কোথায় একটা অ্যারে ব্যবহার না করে ডাটা স্টাকচার এবং অ্যালগোরিদম এর জ্ঞান কাজে লাগিয়ে একটি অ্যারে ব্যবহার করেই কাজ সম্পন্ন করছেন, কিভাবে আপনার কোডের পার্ফর্মেন্স বৃদ্ধি করছেন এগুলোও বিবেচ্য বিষয়। আর কোডের সৌন্দর্য বজায় রেখে কোড তখনই করা সম্ভব যখন আপনার হাজার হাজার ঘণ্টা কোড লেখার এক্সপেরিয়েন্স থাকবে। আর এখানেই দরকার হয় অনলাইন জার্জের।

যে কোনো অনলাইন জার্জ হাজার হাজার প্রশ্নে লিস্টিং করা থাকে, দক্ষতার ওপরে ভিত্তি করে ক্যাটাগরাইজ করা থাকে। অনলাইন জার্জের একটা বড় সুবিধা হচ্ছে এখানে একটা প্রশ্নে এর শুধু সমাধান করলেই হয় না। কোডটা অপ্টিমাইজড কিনা, নির্দিষ্ট সময়ের ভিতরে, নির্দিষ্ট মেমোরি রেঞ্জে এক্সিকিউট হয় কি না, কোডের প্রেসেন্টেশন ঠিক আছে কিনা, হাজার

---

হাজার ভিন্ন ভিন্ন ইনপুটের জন্য কোডটি কাজ করে কিনা সব কিছুই বিবেচনায় আনা হয়। তাই একটা প্রশ্নেম সমাধান করতে অন্ততপক্ষে দশটা ভিন্ন ভিন্ন ওয়েভে আপনাকে চিন্তা করা শিখতে হয়। তারপরেও অনেক সময় কোডটা রিজেকশনের ঝুলিতেই স্থান পায়।

আপনার কোড যদি অনলাইন জাজ রিজেক্ট করে দেয়, মোটেও ভয় পাবেন না। প্রশ্নেম সল্লিং এর জগতে এটা খুবই কমন একটা ব্যাপার। আমি যখন প্রশ্নেম সল্লিং করতাম তখন শুরুর দিকে সাধারণ যোগ বিয়োগের সমাধানও ১৫-২০ বার করে রিজেক্ট পেতাম। খুবই বিরক্ত লাগতো তখন। নিজের চুল ছিঁড়তে মন চাইতো। হাজার খুঁজেও আমি কোনো ভুল পাচ্ছি না, অথচ কোড একসেপ্টেড হচ্ছে না। সব শেষ দেখা যেত আউটপুটের শেষে একটা ইনভিসিবল স্পেস আছে। যার কারণে প্রেসেন্টেশন ইরোর হচ্ছে। এই রকম অসংখ্য ইরোর আপনি পাবেন। তবে ইরোর দখে ভয় পাওয়া যাবে না। ইরোর হচ্ছে আপনার বন্ধু, যে আপনাকে সব রকম ভুল থেকে বাঁচাবে। ছোটোখাটো কোনো ভুল হওয়ার আগেই সে আপনাকে মেসেজ দিয়ে বলবে এটা ঠিক হয়নি, তাড়াতাড়ি ঠিক করো। আমাদের জীবনেও এইরকম একজন বন্ধু দরকার যে আমাদের সমস্ত ভুল গুলো ধরিয়ে সঠিক কাজটা করিয়ে নিতে পারে। তাই আজকে থেকে ইরোরকে ভয় না পেয়ে ইরোরের সাথে বন্ধুত্ব করে নেবেন।

প্রশ্নেম যে শুধু অনলাইন জাজেই দেওয়া থাকে এমনটা না। আপনি যে কোনো জায়গা থেকে প্রশ্নেম নিয়ে সমাধান করতে পারেন। আমি প্রথম দিকে জানতামই না যে অনলাইন এ এইরকম সার্ভিস আছে যেখানে প্রচুর প্রশ্নেম আগে থেকেই দেওয়া থাকে। আমি যেটা করতাম বিভিন্ন সফটওয়্যারের ছোট ছোট ফাংশনালিটিস খুঁজে বের করতাম এবং সেগুলোকে নিজের মত করে তৈরি করার চেষ্টা করতাম। কাজ কিন্তু একই, আপনাকে প্রোগ্রামিং ল্যাংগুয়েজ ব্যবহার করে বাস্তব জীবনের সমস্যা গুলোরই তো সমাধান খুঁজে বের করতে হবে, তাই না? তবে অনলাইন জাজের একটা বড় সুবিধা হচ্ছে আপনার স্কিল অনুযায়ী প্রশ্নেম এর ক্যাটাগরি তৈরি করা থাকে, যার ফলে প্রশ্নেম খুঁজে পেতে এবং একই টাইপের অনেক গুলো প্রশ্নেম একসাথে সল্লু করতে সহজ হয়।

প্রোগ্রামিং এর ফান্ডামেন্টাল কনসেপ্ট গুলো মোটামোটি ভাবে আয়ত্ত করার পরেই আপনি যে কোনো একটা অনলাইন জাজে প্রশ্নেম সল্লিং শুরু করতে পারেন। গুগলে সার্চ করলেই অসংখ্য অনলাইন জাজ পেয়ে যাবেন। এখানে প্রশ্নেম সল্লিং এর কোন লিমিট নেই, যখন আপনি মনে করবেন যে, প্রোগ্রামিং ল্যাংগুয়েজ ব্যবহার করে সব রকম কোডই করতে পারছেন,

কখন কোন ফিচারটা ব্যবহার করবেন সেটা ভাবতে হচ্ছে না, নিজে থেকেই বুঝে যাচ্ছেন তখন আপনি পরবর্তী ধাপে পদার্পণ করতে পারেন।

## পরিচিত কিছু অনলাইন জাজ

[URI](#)

[UVA](#)

[Code  
Forces](#)

[Code  
Chef](#)

[Hacker  
Rank](#)

[SPOJ](#)

আমাদের দেশের সফটওয়্যার ইন্ডাস্ট্রি খুবই ছোট। তাই এখানে প্রৱেশ সল্লিং এর দক্ষতার থেকে ডেভেলপমেন্টকে বেশি শুরুত্ব দেওয়া হয়। কে কয়টা অ্যালগোরিদমের ব্যবহার জানে তার থেকে বেশি কয়টা ফ্রেমওয়ার্ক জানে সেই বিষয়ে বেশি শুরুত্ব দেওয়া হয়। কিন্তু আজকে যদি আমরা অ্যালগোরিদমিষ্ট তৈরি না করে শুধু ফ্রেমওয়ার্ক জানা ডেভেলপার তৈরি করতে থাকি, কালকে আমাদের এই ছোট সফটওয়্যার ইন্ডাস্ট্রিটাও থাকবে না। যেখানে গুগল, ফেসবুকের মত কোম্পানিরা প্রতি বছর কম্পিউটিভ প্রোগ্রামিং এর আয়োজন করে সারা বিশ্ব থেকে বাছাই করে বড় বড় প্রৱেশ সল্লারকে তাদের দলে যুক্ত করছে, সেখানে আমরা প্রৱেশ সল্লিং এর জ্ঞানকে পাওয়ায় দিচ্ছি না। যা আমাদের দেশে সফটওয়্যার ইন্ডাস্ট্রি গড়ে ওঠার সমস্ত সম্ভবনাকে ধূলিসাং করে দিচ্ছে, সমস্ত রাস্তা বন্ধ করে দিচ্ছে। এর ফল ভোগ করতে হবে আমাদের ভবিষ্যৎ প্রজন্মকে।

### যে বিষয় গুলো মনে রাখতে হবে



- ❖ ফান্ডামেন্টালস বিষয় শেখার পরেই প্রৱেশ সল্লিং শুরু করা যায়
- ❖ অনলাইন জাজে প্রৱেশ সল্লু করলে কোডিং দক্ষতা বৃদ্ধি পায়
- ❖ যে কোনো ল্যাংগুয়েজ ব্যবহার করেই প্রৱেশ সল্লু করা যায়
- ❖ সি, সি++ এবং জাভা প্রৱেশ সল্লিং এর জন্য ভালো

**বিঃদ্রঃ** প্রৱেশ সল্লিং করার পূর্বে ক্লাস ৬-১০ পর্যন্ত গণিত বইয়ের বিষয়বস্তু একবার ঝালাই করে নিবেন।

---

এত দিন আমরা কম্পিউটারের সাথে বন্ধুত্ব করার জন্য, তার সাথে কথা বলার জন্য, তাকে দিয়ে নিজের কাজ করিয়ে নেওয়ার জন্য অনেক পরিশ্রম করেছি, তার ভাষা শিখেছি। এখন আমরা তাকে কমান্ড দিয়ে ম্যাথমেটিক্সের হোম ওয়ার্ক গুলো করিয়ে নিতে পারব। কিন্তু সমস্যা হচ্ছে বড় বড় প্রশ্নের আসলে আমরা নিজেরাই বিপাকে পরে যাচ্ছি। কম্পিউটার কি কাজ করবে, তার থেকে তো বেশি কাজ আমাদেরই করতে হচ্ছে। এই সমস্যার সমাধান করতে চাইলে আমাদের হাই লেভেলের ভাষা শিখতে হবে। সাথে সাথে কিছু হাই লেভেলের কমান্ডও শিখে নিতে হবে। যেন আমরা আমাদের বোকা বাক্সটাকে দিয়ে সর্বোচ্চ কাজ করিয়ে নিতে পারি এবং আমাদেরও কম কষ্ট করতে হয়।

নতুন ভাষা শেখার কথা শুনলেই মনের কোনো এক জায়গা থেকে ভয় উঁকি দিয়ে বলে, কোনো ভাবে তা একটা ল্যাংগুয়েজের কিছুটা শিখেছি, আবার নতুন একটা শিখতে হবে? আমার দ্বারা হবে না। বিশ্বাস করেন, আপনার দ্বারাই হবে। আপনি যদি প্রোগ্রামিং এর ফান্ডামেন্টাল বিষয় গুলো ভাল করে আয়ত্ত করে থাকেন, তাহলে নতুন ল্যাংগুয়েজ শেখা আপনার জন্য একদম কঠিন কিছু হবে না। ফান্ডামেন্টালস গুলো সব ল্যাংগুয়েজেই একই।

আমাদের বাস্তব জীবনের সমস্যা গুলো প্রোগ্রামিং ল্যাংগুয়েজ ব্যবহার করে সমাধান করার বিভিন্ন রকমের টেকনিক আছে, বিভিন্ন রকমের প্রোগ্রামিং থিওরি আছে। এগুলোকে বলা হয় Programming Paradigm. এত দিন আমরা যেই প্রোগ্রামিং শিখলাম সেটা হচ্ছে Procedural Paradigm, যার মানে হচ্ছে ওপর থেকে নিচ পর্যন্ত যা কোড আছে সব লাইন বাই লাইন এক্সিকিউট হতে থাকবে। প্রথমে শুনতে খুব ভালই লাগে যে তাই তো, কোড যা লিখব তা লাইন বাই লাইন এক্সিকিউট হবে, এটাই তো সহজ। কিন্তু যখন আপনি বড় বড় প্রজেক্ট করবেন তখন দেখবেন যে এই ভাবে কোড ম্যানেজ করা খুব কষ্টসাধ্য হয়ে যাচ্ছে। আমার আপনার আগেই বড় বড় প্রোগ্রামাররা এই রকম সমস্যার সম্মুখীন হয়েছিলেন। তাই তারা নতুন ধরনের একটা প্রোগ্রামিং টেকনিক, নতুন Programming Paradigm এর জন্ম দিলেন। যার নাম অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং (Object Oriented Programming)। এটা শুনতে খুব ভয়ংকর মনে হলেও বাস্তবে কঠিন কিছু না। পৃথিবীতে প্রচুর অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ আছে, কিন্তু সব জায়গাতেই অবজেক্ট অরিয়েন্টেডের মূল থিওরি একই। আর এবার আমাদের অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং শিখতে হবে।

# Programming Paradigm সম্পর্কে কিছু কথা

**Programming Paradigm** হচ্ছে প্রোগ্রামিং ল্যাংগুয়েজ গুলোকে তার ফিচারের ওপরে ভিত্তি করে আলাদা করার টেকনিক। আপনি এটাকে প্রব্লেম সল্ভ করার মেথড হিসেবে চিন্তা করতে পারেন। একেকটা প্রোগ্রামিং ল্যাংগুয়েজ এক এক ভাবে প্রব্লেম সল্ভ করে থাকে। প্রতিটা প্রোগ্রামিং ল্যাংগুয়েজেই নিজস্ব Paradigm আছে। আবার একটা প্রোগ্রামিং ল্যাংগুয়েজে একাধিক Paradigm মেনে কোড করা যেতে পারে।

সাধারণত Programming Paradigm কে আমরা দুইভাগে ভাগ করতে পারি।

**Imperative:** এই ক্ষেত্রে প্রোগ্রামার কম্পিউটারকে ইন্সট্রাক্ট করে কিভাবে একটা সমস্যার সমাধান করা যায়। Imperative Paradigm এর অন্তর্ভুক্ত Paradigm গুলো হচ্ছে -

- **Procedural:** এখানে অনেক গুলো ইন্সট্রাকশন গ্রহণ আকারে থাকে এবং একটার পরে একটা এক্সিকিউট হতে থাকে।
- **Object Oriented:** এখানেও ইন্সট্রাকশন গুলো গ্রহণ আকারেই থাকে, তবে এখানে ইন্সট্রাকশনের সাথে সাথে state বা ভ্যালুও বাইন্ডিং অবস্থাতে থাকে।

**Declarative:** এই ক্ষেত্রে প্রোগ্রামারকে শুধুমাত্র বলে দিতে হয় যে সে কি আউটপুট চাচ্ছে। কোনো রকম কম্পিউটিং লজিক নিয়ে কাজ করতে হয় না।

- **Functional:** প্রোগ্রামার যেই আউটপুট টা পেতে চাচ্ছে সেই আউটপুটটা পাওয়ার জন্য তাকে অনেক গুলো ফাংশন আকারে তার কাংখিত আউটপুটটা লিখতে হবে।

বিস্তারিত জানুন



---

আপনি যে কোনো একটা ল্যাংগুয়েজ যা কিনা অবজেক্ট অরিয়েন্টেড (Object Oriented) সাপোর্ট করে, বেছে নিতে পারেন আপনার অবজেক্ট অরিয়েন্টেড হাতিয়ার হিসেবে। তবে আমি বেছে নিব জাভা। এর পিছনে দুইটি কারণ আছে। প্রথম কারণ হচ্ছে এটি জন্মগতভাবেই অবজেক্ট অরিয়েন্টেড। মানে এখানে একলাইন কোডও যদি আপনি লিখতে চান, আপনাকে অবজেক্ট অরিয়েন্টেডের থিওরি মনেই লিখতেই হবে। আর দ্বিতীয় কারণ হল, আমি জাভাকে খুব ভালবাসি। আমার রক্ষের অণুতে অণুতে জাভার সিনট্যাক্স মিশে আছে। এন্টারপ্রাইজ অ্যাপ্লিকেশনের দুনিয়াতে এখনো জাভারই রাজত্ব চলে। তাই আমি মনে করি, সি শেখার পরে জাভাতে হাত দেওয়াটাই বুদ্ধিমানের কাজ। আর আপনি যদি সি++, সি শার্প বা পাইথন শিখতে চান তাহলেও কোন সমস্যা নেই। তবে ডেভেলপার হিসেবে নিজের ভিত্তি স্থাপন করার জন্য জাভা এর থেকে ভালো ল্যাংগুয়েজ আছে বলে আমার মনে হয় না। যখন আর কোনো কাজে জাভা ব্যবহৃত হবে না, তখনও নিজের বেস তৈরির অস্ত্র হিসেবে ব্যবহার করা হবে এই জগৎবিখ্যাত ল্যাংগুয়েজটিকে। যদি আপনার জাভা সম্পর্কে নূন্যতম একটা জ্ঞান না থাকে, তাহলে দুনিয়ার শ্রেষ্ঠ বই গুলো পড়ার মজা থেকে আপনি বঞ্চিত হবেন।

জাভা শুরু করার পূর্বেই জাভা সম্পর্কে বেসিক কিছু তথ্য অনলাইন ঘুঁটে জেনে নেওয়াটা বুদ্ধিমানের কাজ হবে, তারপরে এর জন্য এনভাইরনমেন্ট সেটআপ করাটা শিখতে হবে। এর পরে কিছুদিন প্রোগ্রামিং ফার্মান্টালস গুলোই অনুশীলন করতে হবে, কিন্তু এবার কোড গুলো লিখবেন জাভা ব্যবহার করে। যেহেতু সি জাভা প্রোগ্রামিং এর ও মা, তাই সিনট্যাক্সগত খুব একটা পরিবর্তন আপনি লক্ষ্য করবেন না। পরিবর্তন আসবে ডাটা টাইপস, অপারেটরসএ, ফাংশন ডেফিনিশনে। এই ছোট ছোট পরিবর্তন গুলো আপনাকে খুব বেশি সমস্যায় ফেলবে না, অল্প একটু হাঁটাধাঁটি করলেই সি এর মত জাভাতেও আপনি কম্পিউটারের সাথে কথা বলতে পারবেন।

যখন মনে হবে আপনি জাভাতে কোড লিখতে পারছেন, পূর্বের সি ব্যবহার করে সল্লু করা প্রশ্নে গুলো জাভাতেও সল্লু করতে পারছেন, তখন একটু সামনে আগাতে হবে। যেই উদ্দেশ্যে জাভা প্রোগ্রামিং শিখতে চাওয়া এবার সেই পথে হাঁটা শুরু করতে হবে। মানে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং শেখা শুরু করতে হবে। যদিও জাভাতে প্রথম দিন থেকেই অবজেক্ট অরিয়েন্টেড ওয়েতেই কোড করতে হয়, তারপরও এবার আপনি সত্যিকার অর্থেই অবজেক্ট অরিয়েন্টেড শেখা শুরু করবেন।

---

**অবজেক্টঃ** সবার প্রথমে আপনাকে জানতে হবে অবজেক্ট কাকে বলে, কিভাবে একটা অবজেক্ট খুঁজে বের করতে হয়? যে কোন কিছু থেকে অবজেক্ট খুঁজে বের করতে পারলেই ধরে নিবেন, অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ৫০% আপনার বোঝা হয়ে গেছে। আমি যখন প্রথম অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং শেখা শুরু করি তখন কোনভাবেই অবজেক্ট খুঁজে পেতাম না। যার ফলে পুরো কনসেপ্টটাও সঠিক ভাবে বুঝতে পারতাম না। সহজ কথায় যদি বলি, যে কোন কিছু যাকে সঠিক ভাবে বর্ণনা করার জন্য একাধিক বিষয় দরকার হয় সেটাই অবজেক্ট। যেমন, মানুষ। আপনি শুধু মানুষ বললে আমরা সঠিক কোন ইনফরমেশন পাব না। আপনাকে বলতে হবে, মানুষের নাম কি, তার বাবার নাম কি, তার ঠিকানা কি, তার উচ্চতা কত, গায়ের রং কি আরও কত কিছু। এত এত বিষয় যখন আপনি বলবেন ঠিক তখনই আমরা বুঝতে পারবো যে আপনি আসলে কোন মানুষটার কথা বলছেন। তাহলে এখানে একটা মানুষকে সঠিক ভাবে বর্ণনা করতে আমাদের কত গুলো বিষয় তুলে আনতে হল? একারণেই মানুষ একটা অবজেক্ট। এবার আপনি আপনার চারপাশে একটু খুঁজে দেখেন তো আর কি কি অবজেক্ট পান? অসংখ্য অবজেক্ট পাবেন, আসলে আমরা অবজেক্ট দ্বারাই ঘিরে আছি।

**ক্লাসঃ** অবজেক্ট হচ্ছে একধরনের ডাটা স্ট্রাকচার, যেখানে আমরা একটা নির্দিষ্ট বস্তুর অনেক রকম ডাটা একসাথে স্টোর করে রাখি। আমাদের এই অবজেক্ট তৈরি করার জন্য একটা ক্লাসিন্ট দরকার, যেই ক্লাসিন্ট দেখে আমরা হাজার হাজার অবজেক্ট তৈরি করতে পারি। আর এই ক্লাসিন্টই হচ্ছে ক্লাস। একটা ক্লাসের ভিতরে একবার আমাদেরকে বলে দিতে হবে যে অবজেক্টটা দেখতে কেমন হবে, তারপর হাজার বার আমরা এই ক্লাস ব্যবহার করে নতুন নতুন অবজেক্ট তৈরি করতে পারব। এখানে প্রতিটো অবজেক্ট কিন্তু একই ডাটা ধরে রাখবে না, সবাই ভিন্ন ভিন্ন ডাটা ধরে রাখবে যদিও সব গুলো অবজেক্টই তৈরি হয়েছে একটা মাত্র ক্লাস থেকে।

**ইনহেরিটেন্সঃ** ক্লাস এবং অবজেক্ট মোটামোটি আয়ত করার পরেই চলে আসবে ইনহেরিটেন্স, যার মানে হল অন্যের বৈশিষ্ট্য নিজের ভিতরে ধারণ করা। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর একটা মূল লক্ষ্য হল কোড ডুপ্লিকেশন না করা। আর কোড ডুপ্লিকেশন এড়ানোর জন্যই ইনহেরিটেন্সের আবির্ভাব, যেখানে অন্যের কাছে থেকে ধার নিয়ে চলবে সেখানে নতুন করে কোড লেখার কোন দরকারই নেই। এই থিওরিটো DRY নামে পরিচিত যার মানে হচ্ছে Do Not Repeat Yourself. কোডের সৌন্দর্য এবং রক্ষণাবেক্ষণ নিশ্চিত করতে আমাদের যতটা সম্ভব কোড ডুপ্লিকেশন এড়িয়ে চলতে হবে।

**পলিমোরফিসমঃ** এর অর্থ হল বহুরূপী। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এ একটা অবজেক্ট, একটা মেথড বা একটা ভ্যারিয়েবলও বহুরূপী আচরণ করতে পারে। এই বহুরূপী আচরণের মূল কারণও কিন্তু কোড ডুপ্লিকেশন এড়ানো। এখানে একই নামের ভিন্ন ভিন্ন মেথড থাকতে পারে। প্রয়োজনের ওপরে ভিত্তি করে একই মেথড ভিন্ন ভিন্ন আচরণ করতে পারে। অবজেক্টের ওপরে, আর্গুমেন্ট এর ওপরে নির্ভর করবে কখন কোন মেথডটি এক্সিকিউট হবে। যার ফলে অনেক লজিক্যাল অপারেশন আমাদের নিজে থেকে করার প্রয়োজন হয় না।

**নোটঃ** যখন একটা ফাংশন কোনো একটা ক্লাসের ভিতরে ডিফাইন করা হয় অথবা যখন একটা অবজেক্ট এর অংশ হয় তখন সেই ফাংশনকে মেথড বলা হয়।

**ইনক্যাপ্সুলেশন এবং অ্যাবস্ট্রাকশনঃ** অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর একটা বড় বৈশিষ্ট্য হল, জটিল জটিল কোড গুলো নিজের কাছে লুকিয়ে রাখা এবং অন্ন কয়েকটা ফাংশন বা মেথড বাইরে এক্সপোস করা। যেন এই অবজেক্টের ব্যবহারকারী এই জটিল কোডগুলো সম্পর্কে কোনভাবেই জানতে না পারে। জাভাতে এই কাজ গুলো করার জন্য এক্সেস মোডিফায়ার রয়েছে, রয়েছে অ্যাবস্ট্রাক্ট ক্লাস এবং ইন্টারফেস।

আসলে অবজেক্ট অরিয়েন্টেড খুব বড় সড় কোন থিওরি না। ছোট থিওরি হলেও এটার ব্যবহার বুঝতে এবং প্রলৈম অনুযায়ী সঠিকভাবে এর থিওরি অ্যাপ্লাই করা শিখতে কিছুটা সময় লেগে যায়। এর জন্য বিভিন্ন ম্যানেজমেন্ট সিস্টেম টাইপের ছোট ছোট প্রজেক্ট করা যতে পারে। সাথে ডিজাইন প্যাটার্ন গুলো সম্পর্কেও ধারণা নেওয়া যতে পারে। তবে আপনি যদি কেবল মাত্র অবজেক্ট অরিয়েন্টেড সম্পর্কে ধারণা লাভ করেন তাহলে আমি বলবো ডিজাইন প্যাটার্ন আরও কয়েকদিন পরেই হাত দিন।

## Four Pillar of Object Oriented Programming

Encapsulation

Abstraction

Inheritance

Polymorphism

অবজেক্ট অরিয়েন্টেড বাদেও জাভার কিন্তু একটা বিশাল লাইব্রেরী রয়েছে। যেখানে প্রচুর প্যাকেজ এবং ক্লাস রয়েছে, প্রতিনিয়ত আমরা যেধরনের সমস্যার সম্মুখীন হয়, তার প্রায় সমস্ত সমস্যারই সমাধান আগে থেকেই রয়েছে। যদি কেউ মনে করে যে আমি প্রতিদিন একটা করে বিউল্টইন ক্লাস শেষ করবো তাহলে তার ৩৬০০+ দিন লাগবে। আসলে জাভার সব গুলো ক্লাস শেখার আপনার দরকারই নেই, যখন যেটা দরকার হবে শিখে নেওয়া যাবে। শুধু আপনাকে শিখতে হবে, কিভাবে আপনি জাভার অফিশিয়াল ডকুমেন্টেশনটা ফলো করবেন। একবার ডকুমেন্টেশনটা বুঝতে পারলে যখন যেই প্যাকেজ বা ক্লাসই আপনার দরকার হোক না কেন আপনি অনায়াসে সেটা ব্যবহার করতে পারবেন।

আমার মনে হয় অন্তত পক্ষে একবছর শুধু কোর জাভা এবং এর বিউল্টইন লাইব্রেরী নিয়েই ঘাঁটাঘাঁটি করা উচিত। অনেক বিগিনার অল্প কয়েকদিন জাভা অনুশীলন করেই ডেভেলপমেন্টে জাম্প করে, এবং পরে প্রোগ্রামিং ল্যাংগুয়েজ, এখানে শেখার মতো অনেক কিছু আছে। আপনি এটা ধরে নিতেই পারেন যে, জাভা সঠিকভাবে শিখতে পেরেছেন মানে আপনার ভবিষ্যৎ উজ্জ্বল। আপনি ভবিষ্যতে জাভা ব্যবহার করবেন কি করবেন না এটা গুরুত্বপূর্ণ না। কিন্তু জাভা আপনার যেই ভিত্তি গড়ে দিবে, তার কাছে আপনি সারা জীবন কৃতজ্ঞ থাকতে বাধ্য থাকবেন। আমি আমার লাইফে চার বছর জাভা নিয়ে ছিলাম। এখন জাভা নিয়ে সরাসরি কোনো কাজ করিনা। কিন্তু জাভার প্রতি আমার কৃতজ্ঞতা সারা জীবন থাকবে।

## প্রথম ল্যাংগুয়েজ হিসেবে জাভা

প্রোগ্রামিং এর কনসেপ্ট আমি শিখি জাভা এর কাছেই এবং জাভা আমার প্রথম প্রোগ্রামিং ল্যাংগুয়েজ ছিল। জাভা দিয়ে আপনি আপনার প্রোগ্রামিং ক্যারিয়ার শুরু করতেই পারেন। কিন্তু সেই ক্ষেত্রে কিছু বিষয় অজানা থেকে যাওয়ার সম্ভাবনা থেকেই যায় যেহেতু জাভা একটি হাই লেভেল ল্যাংগুয়েজ। জাভা দিয়ে শুরু করলেও খুব বেশি বিপদে আপনাকে পড়তে হবে না।

তবে আমার সাজেশন থাকবে যদি সম্ভব হয় তাহলে সি প্রোগ্রামিং ল্যাংগুয়েজ দিয়েই শুরু করবেন। এতে প্রতিটা বিষয় বুঝতে জাভার থেকে অনেক বেশি সহজ হবে।

## কেন জাভা শিখা উচিত

আপনি যদি গুগলে ‘Why Should We Learn Java’ লিখে সার্চ করেন তাহলে হাজার হাজার আর্টিকেল এবং ভিডিও দেখতে পারবেন। তারা যা বলবে তার বেশির ভাগই সঠিক তথ্য। জাভা কেন শিখবো এই প্রশ্নের উত্তরের কোনো অভাব নেই। তবে জাভা কেন শিখবো না এই প্রশ্নের উত্তর যদি খুঁজে পান তাহলে আপনার নোবেল পাওয়ারও সম্ভবনা আছে। আমি এখানে আমার নিজস্ব মতামত গুলোই দেওয়ার চেষ্টা করবো। তবে কিছু কিছু তো অনলাইন এর সাথে মিলেই যাবে।

- হাই পার্ফরমিং অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ গুলোর ভিতরে সি++ এর পরেই জাভার অবস্থান। সি++ এর লার্নিং কমপ্লেক্সিটি বেশি হওয়ায় এন্টারপ্রাইস অ্যাপলিকেশনের একরকম স্ট্যান্ডার্ড হয়ে গিয়েছে জাভা। জাভা ব্যবহৃত হয় না এমন কোনো ক্ষেত্র খুঁজে পাওয়াই মুশকিল। আপনার ঘরের ব্যবহৃত গ্যাজেট থেকে শুরু করে প্রতিদিন ব্যবহার করা ইউটিউব নেটফ্লিক্সের ব্যাকেন্ডও ব্যবহৃত হচ্ছে জাভা।
- বর্তমানে জাভার পপুলারিটি হারিয়ে যাচ্ছে বলে একটা হাইপ তৈরি হলেও বছরের পর বছর শীর্ষে অবস্থান করা এই ল্যাংগুয়েজটি এখনো বিশ্বের সেরা তিনটি প্রোগ্রামিং ল্যাংগুয়েজের একটা। কম্পিউটার সাইন্সের অন্যতম ভিত্তি হচ্ছে এই ল্যাংগুয়েজটি। কম্পিউটার সাইন্স, সফটওয়্যার আর্কিটেকচার, এন্টারপ্রাইস অ্যাপলিকেশনের ওপরে সারা বিশ্বে যত নাম করা বই আছে, তার বেশির ভাগই লেখা হয়েছে জাভার ওপরে ভিত্তি করে। অন্ততপক্ষে এই বই গুলোর স্বাদ উপভোগ করার জন্য হলেও, বই গুলোর নির্জন পেতে হলেও আপনাকে জাভা জানতে হবে।
- জাভা হচ্ছে একটা পিউর অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ। অবজেক্ট অরিয়েন্টেডের যেই স্পেসিফিকেশন আছে তার প্রোপার ইমপ্লিমেন্টেশন পাওয়া যায় জাভাতে। তাই আপনি যদি সত্যিকার অর্থেই অবজেক্ট অরিয়েন্টেড শিখতে চান, ডিজাইন প্যাটার্ন গুলোর ইমপ্লিমেন্টেশন ভালো ভাবে বুঝতে চান তাহলে আপনার জন্য সব থেকে ভালো সাজেশন হল জাভা শিখুন।

## কেন জাভা শেখা উচিত

- জাভা সি প্রোগ্রামিং ল্যাংগুয়েজের মত এতটাও লো লেভেল না আবার জাভাস্ক্রিপ্টের মত এতটাও হাই লেভেল না। তাই এখানে আপনাকে পয়েন্টার, মেমরি ম্যানেজমেন্ট, গার্বেজ কালেকশনের মত বিষয়গুলো নিয়ে কাজ করতে হবে না। এই কাজ গুলো কম্পাইলার আপনার হয়ে করে দেবে। কিন্তু জাভা স্ট্যাটিক টাইপ ল্যাংগুয়েজ হওয়ায় আপনাকে ডাটা টাইপ নিয়ে কাজ করতে হবে যার ফলে আপনার কোড অনেক বেশি ম্যানেজেবল হবে।
- জাভার রয়েছে বিরাট এক স্টার্ডার্ড লাইব্রেরী কালেকশন, বিরাট বড় কমিউনিটি। জাভা শেখাও খুব সহজ কারণ জাভার সাথে রয়েছে একটা রিচ ডকুমেন্টেশন। এছাড়াও যেকোনো ডেভেলপমেন্টের ক্ষেত্রেই জাভার ইকো সিস্টেম অন্য যেকোনো ল্যাংগুয়েজের থেকে বড়। কারণ জাভা বহু বছর ধরেই নিজের অবস্থান মার্কেটের শীর্ষেই ধরে রেখেছে।

### Widely Used Applications Of Java



# জাভা প্রোগ্রামিং এ যা শিখতে হবে

## Step 1 - Environment Setup

- Java Virtual Machine
- Java Development Kit
- Install Eclipse or IntelliJ Idea

## Step 2 - Basic Syntax

- Variables and Data Types
- Conditionals and Functions
- System IO
- Array and String
- Fundamental Topics

## Step 4 - OOP Practice

- Date Classes
- Calendar Classes
- Wrapper Classes

## Step 6 - Exception Handling

- Understanding Exception
- Built-in Exceptions
- Custom Exception

## Step 7 - GUI Programming

- Java Swing
- Java FX

## Step 9 - Synchronisation

- Synchronised Block
- Inter-Thread Communication
- Deadlock and Interrupting Thread

## Step 3 - OOP

- Understand OOP
- Class and Object
- Inheritance
- Polymorphism
- Method Overloading
- Method Overriding
- Encapsulation
- Access Modifier
- Abstract Class & Method
- Interface

## Step 5 - Collection Framework

- Generics in Java
- ArrayList & LinkedList
- HashSet & TreeSet
- Queue & Priority Queue
- Dequeue
- Map, HashMap & TreeMap
- Collection and Sortable
- Comparable and Comparator

## Step 8 - Concurrency

- Understanding Thread
- Thread Pool, Thread Group
- Thread Priority and Joining
- Multiple Tasks
- Garbage Collection
- Runtime Class

**And Related lot of Topics...**

## জাভা প্রোগ্রামিং শিখার রেফারেন্স

### জাভা প্রোগ্রামিং এর বই সমূহঃ

- Java: How to Program by Deitel, Deitel
- Head First Java by Bert Bates
- Java Complete Reference By Herbert Schildt
- Thinking in Java By Bruce Eckel
- Head First Object Oriented Analysis & Design by Brett McLaughlin
- Head First Design Patterns By Freeman

### জাভা প্রোগ্রামিং এর ইউটিউব চ্যানেলঃ

- [Anisul Islam](#) (Bangla)
- [Stack Learner](#) (Bangla)
- [Telusko Learning](#) (English)
- [Edureka](#) (English)

### জাভা প্রোগ্রামিং ওয়েবসাইটঃ

- [Geeks For Geeks](#)
- [Programmiz](#)
- [Beginners Book](#)
- [Learn Java Online](#)
- [Oracle Official Tutorial](#)
- [Java Official Documentation](#)

### জাভা প্রোগ্রামিং ট্রেনিং প্রোগ্রাম এবং কোর্সেসঃ

- [Stack Learner Premium Courses](#)
- [Stack Learner Bootcamps \(Training Program\)](#)

বিঃদ্রঃ একটি নির্দিষ্ট সময়ে একটি নির্দিষ্ট রিসোর্স মনে চলবেন

## জনপ্রিয় কিছু কোড এডিটর এবং IDE



## জাভা সম্পর্কে কিছু গুরুত্বপূর্ণ কথা

**Java SE:** যখন আমরা জাভা সম্পর্কে শুনে থাকি তখন আমরা জাভার এই এডিশন সম্পর্কেই চিন্তা করি। Java SE এর পূর্ণরূপ হচ্ছে Java Standard Edition. প্রথমে আপনাকে এই এডিশনটাই শিখতে হবে। কারণ জাভা বলতে যা বোঝায় তা হচ্ছে জাভার স্ট্যান্ডার্ড এডিশন। অ্যান্ড্রয়েড অ্যাপলিকেশন তৈরিতে জাভার এই ভার্সনই ব্যবহার করা হয়। আপনি যে কোনো কাজই এই জাভার স্ট্যান্ডার্ড এডিশন ব্যবহার করে করতে পারবেন।

**Java EE:** এন্টারপ্রাইস অ্যাপলিকেশনের জন্য আলাদা ভাবে ডিজাইন করা এই এডিশনের পূর্ণ রূপ হচ্ছে Java Enterprise Edition যা মূলত স্ট্যান্ডার্ড এডিশনের ওপরে ভিত্তি করেই তৈরি করা হয়েছে। তবে এন্টারপ্রাইস এডিশন ব্যবহার করা হয় বড় এন্টারপ্রাইস অ্যাপলিকেশনের ক্ষেত্রে যেখানে মাল্টি প্রেডের প্রচুর কাজ থাকে, নেটওয়ার্ক রিকুয়েষ্টের কাজ থাকে এবং যেই অ্যাপলিকেশন গুলো প্রচুর স্কেল করার দরকার হয়। আপনি যদি জাভার স্ট্যান্ডার্ড এডিশন সম্পর্কে জ্ঞান বাখেন তাহলে খুব কম সময়েই আপনি এন্টারপ্রাইস এডিশনও কাজ করতে পারবেন।

**বিঃদ্রঃ** জাভা স্ট্যান্ডার্ড এডিশন ভালোভাবে আয়ত্ত করার পূর্বে ভুল করেও এন্টারপ্রাইস এডিশন শেখার চেষ্টা করবেন না। জাভা বলতে এর স্ট্যান্ডার্ড এডিশনকেই বোঝায়।



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>



---

অধ্যায় তিন

## ভাষার ব্যাকরণ

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ ডাটা স্ট্রাকচার এর গুরুত্ব
- ✓ কমন ডাটা স্ট্রাকচারস
- ✓ অ্যালগোরিদমের গুরুত্ব
- ✓ কমন অ্যালগোরিদমস
- ✓ ডিসক্রিট ম্যাথেমেটিক্স
- ✓ ডিসক্রিট ম্যাথেমেটিক্সের গুরুত্ব

---

কম্পিউটার সাইন্সের সব থেকে গুরুত্বপূর্ণ দুইটি বিষয় হচ্ছে ডাটা স্ট্রাকচার এবং অ্যালগোরিদম। এত দিন তো আমরা অনেক ভাবেই কম্পিউটারের সাথে কথা বলা শিখেছি। কম্পিউটারকে নিজের বন্ধু বানানোর জন্য তার ভাষা শিখেছি। এখন আমরা শুধু কম্পিউটারকে নির্দেশ দিব, আর কম্পিউটার সেই অনুযায়ী কাজ করবে। কিন্তু এই নির্দেশ দেওয়ার ভিতরেও অনেক বিষয় আছে। আমাদেরকে এমন ভাবে নির্দেশ দিতে হবে যেন কম্পিউটার খুব কম সময়েই সব কাজ করতে পারে। আসলে দেখে মনে হয়, এই বোকা বাত্রাটা চুটকিতেই সব কিছু করে ফেলছে। কিন্তু যখন অনেক বড় বড় সব জটিল ক্যালকুলেশন করতে দেওয়া হয়, যখন কোটি কোটি ডাটার ভিতর থেকে একটা ডাটা খুঁজে বের করতে বলা হয়, তখন কিন্তু আমাদের কম্পিউটার মহাশয় একটু ঘাবড়ে যায়। তখন তার কাজ করতে অনেক সময় লাগে। তাই আমাদেরকে ডাটা গুলো রাখার সময় এমনভাবে রাখতে হবে, যেন কম্পিউটার এক চুটকিতেই সব কিছু খুঁজে বের করতে পারে কয়েক ন্যানো সেকেন্ডের মধ্যেই।

কম্পিউটারের জগতে আমরা যত সফটওয়্যার দেখে থাকি, সেটা হোক ব্যাংকের হিসাব রক্ষক সফটওয়্যার অথবা হোক কোন মাল্টিপ্লেয়ার কমব্যাট গেম, আমরা মূলত ডাটা নিয়েই কাজ করে থাকি। কম্পিউটারে সব কিছুই হচ্ছে ডাটা। আর বিভিন্ন সফটওয়্যারের মাধ্যমে আমরা ডাটা ধারণ করা, রক্ষণাবেক্ষণ করা এবং প্রয়োজন মত পরিবর্তন করার কাজ করে থাকি। তাই ভাল মত ডাটার স্ট্রাকচারটা বুঝতে পারা একজন প্রোগ্রামার এবং একজন ডেভেলপার হিসেবে খুবই গুরুত্বপূর্ণ।

ডাটা স্ট্রাকচার মানে হল ডাটাগুলোকে সাজিয়ে গুছিয়ে এমনভাবে রাখা যেন প্রয়োজনের সময় সহজে খুঁজে বের করা যায়, নতুন কোন ডাটা যদি যুক্ত করতে হয় তাহলে খুব সহজেই যেন যুক্ত করা যায় এবং প্রয়োজন না হলে যেন খুব সহজেই কম্পিউটার মেমরি থেকে মুছে ফেলা যায়। এই ডাটা স্ট্রাকচার কিন্তু বল বছর আগে থেকেই আছে, কারণ এটি ম্যাথেমেটিক্সের একটি শাখা। কম্পিউটার সাইন্সের সাথে এর সম্পর্কের বল আগে থেকেই মানুষ এটি ব্যবহার করে আসছে আমাদের দৈনন্দিন জীবনের হিসেব নিকেশের কাজে। কম্পিউটার সাইন্স আসার আগেও মানুষ কোটি কোটি ডাটার হিসেব রেখেছে, তখনও মানুষকে কোননা কোন ভাবে ডাটা গুলোকে গুছিয়ে রাখতে হতো। তফাত হচ্ছে, এখন আমরা সেই পুরোনো সূত্র গুলো নতুন ভাবে ব্যবহার করে বড় বড় সমস্যার সমাধান করাবো আমাদের বোকা মেশিনকে দিয়ে।



## What is Data Structure?

- In computer science, a data structure is a data organisation, management, and storage format that enables efficient access and modification. More precisely, a data structure is a collection of data values, the relationships among them, and the functions or operations that can be applied to the data. ([Wikipedia](#))

একটা সমস্যার যখন আপনি যৌক্তিক সমাধান খুঁজে বের করতে যাবেন তখন প্রথমেই যেই কাজটা দরকার, সমস্যার সাথে সংযুক্ত সকল ডাটা গুলোকে খুঁজে বের করতে হবে। তারপরে ডাটা গুলোকে সুন্দর করে সাজিয়ে ভিন্ন ভিন্ন ডাটার ভিতরের সম্পর্কটা খুঁজে বের করতে হবে। যদি আপনি ডাটা এবং তাদের ভিতরের সম্পর্ক গুলো খুঁজে বের করতে পারেন তাহলেই আপনার সমস্যার বেশির ভাগ সমাধান হয়ে যাবে। কিন্তু আমি দেখেছি, বেশিরভাগ নবীন ডেভেলপাররা শুধুমাত্র ডাটা স্ট্রাকচারের জ্ঞান না থাকার কারণে এই কাজ করতে ব্যর্থ হয়। আপনি ভবিষ্যতে যেই কাজই করেন না কেন, প্রত্যক্ষ বা পরোক্ষ ভাবে ডাটা স্ট্রাকচার এবং অ্যালগোরিদম ব্যবহার করবেন। তাই ভাল হয় ক্যারিয়ারের শুরুতেই বহুল ব্যবহৃত ডাটা স্ট্রাকচারস এবং অ্যালগোরিদম গুলো শিখে নেওয়া। আর একবার শিখে নিতে পারলেই যে কোন সমস্যার সমাধান করা আপনার জন্য কোন ব্যাপারই হবে না।

ডাটা স্ট্রাকচার হল ম্যাথেমেটিক্যাল সেট, ম্যাট্রিক্স, ট্রি, গ্রাফ, বিন্যাস সমাবশের মত বিষয়। এর সাথে প্রোগ্রামিং এর কোনো সম্পর্ক নেই। আপনাকে প্রথমে সমস্ত ডাটা স্ট্রাকচারের থিওরি ভালো ভাবে বুঝতে হবে। কারণ খুব অল্প কিছু সময় আপনাকে নিজে থেকে এই ডাটা স্ট্রাকচার গুলো ইমপ্লিমেন্ট করে কাজ করতে হবে। তবে যখন আপনি শিখবেন তখন নিজের জ্ঞানকে সমন্বয় করতে অবশ্যই যে কোন প্রোগ্রামিং ল্যাংগুয়েজ ব্যবহার করে প্রতিটা ডাটা স্ট্রাকচার ইমপ্লিমেন্ট করবেন। আপনার যদি থিওরি পরিষ্কার থাকে তাহলে যে কোন ল্যাংগুয়েজ ব্যবহার করেই আপনি এগুলোকে ইমপ্লিমেন্ট করতে পারবেন। তবে তার জন্য আপনার প্রোগ্রামিং ল্যাংগুয়েজ এর জ্ঞানটাও ভালো মতই থাকতে হবে।

ডাটা স্ট্রাকচারকে আমরা দুই ভাগে ভাগ করতে করতে পারি। Linear এবং Nonlinear ডাটা স্ট্রাকচার। প্রথম আমরা বহুল ব্যবহৃত লিনিয়ার বা রৈখিক ডাটা স্ট্রাকচার সম্পর্কে জেনে নেই, তারপরে ননলিনিয়ার গুলো সম্পর্কে জানা যাবে।

লিনিয়ার বা রৈখিক ডাটা স্ট্রাকচার গুলো সব থেকে সহজ। এখানে ডাটা গুলোকে একটার পরে আর একটা পাশাপাশি রেখে সংরক্ষণ করা হয়, অনেকটা একটা বই রাখার র্যাকের মত। এই ডাটা স্ট্রাকচার গুলোর একটা নির্দিষ্ট শুরু এবং শেষ আছে। অ্যারে, সেট, লিংকড লিস্ট, স্ট্যাক, কিউই, ম্যাপ এগুলো হচ্ছে সব থেকে বেশি ব্যবহৃত লিনিয়ার ডাটা স্ট্রাকচার। ডাটা স্ট্রাকচার শেখার শুরুর দিকে আমাদের এই ডাটা স্ট্রাকচার গুলোই শিখতে হবে। যেগুলোর ব্যবহার আপনি প্রোগ্রামিং প্রোগ্রামে সল্ভিং এ হরহামেশায় দেখতে পাবেন।

## লিনিয়ার ডাটা স্ট্রাকচারস

### Basic Data Structures

- Array
- Linked List
- Set
- Stack
- Queue

### Advanced Data Structures

- Map
- Doubly Linked List
- Priority Queue
- Dequeue

নন লিনিয়ার ডাটা স্ট্রাকচারগুলো তুলনামূলক কঠিন। কারণ এখানে ডাটার নির্দিষ্ট কোন শুরু বা শেষ থাকে না। যে কোন জায়গা থেকেই ডাটাকে এক্সেস করা যায়। তুলনামূলক জটিল সব সমস্যার সমাধান করতে এই ধরনের ডাটা স্ট্রাকচার গুলো ব্যবহৃত হয়। যেমন পৃথিবীর ম্যাপ ইমপ্লিমেন্ট করার জন্য গ্রাফ ডাটা স্ট্রাকচারটা ব্যবহার করা হয়েছে। সব থেকে জনপ্রিয় নন লিনিয়ার ডাটা স্ট্রাকচার গুলো হচ্ছে ট্রি, হিপ এবং গ্রাফ। একটু কঠিন হলেও আমি মনে করি এই ডাটা স্ট্রাকচার গুলোও আপনার শেখা উচিত। তাহলে আপনার প্রোগ্রাম সল্ভিং সম্পর্কে নতুন একটা ধারণা জন্ম নিবে।

## ননলিনিয়ার ডাটা স্ট্রাকচারস

### Basic Data Structures

- Hash Map
- Hash Table
- Heaps
- Tree
- Binary Search Tree

### Advanced Data Structures

- Multiway Trees
- Trie
- Hash Tree
- Decision Tree
- Graphs

ডাটা স্ট্রাকচার খুব গুরুত্বপূর্ণ বিষয়, কিন্তু ডাটা স্ট্রাকচার একা একাই সব কিছু করতে পারে না। ডাটা স্ট্রাকচারের কথা বললে আপনা আপনিই আর একটা নাম চলে আসে, আর সেটা হল অ্যালগোরিদম। অ্যালগোরিদম কে যদি সহজ ভাবে বোঝাতে চাই তাহলে বলতে হবে একটা সমস্যা সমাধান করার নির্দিষ্ট ধাপ। কোন লাইনের পরে কোন লাইন লিখতে হবে, কোনটার পরে কোনটা এক্সিকিউট হবে, কোথায় কন্ডিশন, কোথায় লুপ হবে, সেগুলোর ধাপে ধাপে বর্ণনা। ডাটা স্ট্রাকচারের সাথে অ্যালগোরিদমের সম্পর্কও কিন্তু এখানেই। আমরা জানি কম্পিউটার ডাটা ছাড়া কিছুই বোঝে না, সে যা করে সব কিছু ডাটা কেন্দ্রিক। সঠিকভাবে ডাটা সংরক্ষণ করার জন্য আমাদের দরকার ডাটা স্ট্রাকচার। আর এই সংরক্ষিত ডাটা খুঁজে বের করা, আপডেট করা, রিমুভ করার জন্য আমরা যেই কোড লিখি সেটাই হল অ্যালগোরিদম।



### What is Algorithm?

- In mathematics and computer science, an algorithm is a finite sequence of well-defined, computer-implementable instructions, typically to solve a class of problems or to perform a computation. Algorithms are always unambiguous and are used as specifications for performing calculations, data processing, automated reasoning, and other tasks. ([Wikipedia](#))

আমরা নিজেরাই প্রতিটা কাজের জন্য অ্যালগোরিদম লিখতে পারি। তবে কিছু সাধারণ প্রব্লেম যেমন - সার্চ করা, সর্ট করা এই ধরনের কাজের জন্য আগে থেকেই খুব সহজ এবং এফিসিয়েন্ট ভাবে অ্যালগোরিদম তৈরি করা আছে যেন কম্পিউটার খুব দ্রুত কাজ শুলো করতে পারে। এই সব অ্যালগোরিদম অনেক বছরের পরীক্ষিত। তাই আমাদের কে প্রথমে এই অ্যালগোরিদম শুলোই শিখতে হবে। পরে প্রয়োজন মত আমরাও অ্যালগোরিদম তৈরি করতে পারব।

প্রথমেই আমাদের শেখা উচিত অ্যালগোরিদম এনালাইসিস করা। কোন অ্যালগোরিদমটার পার্ফরমেন্স কেমন হবে বা আমরা যেই কোডটা লিখলাম সেটা এক্সিকিউট হতে কতটা মেমরি লাগবে, কতটা সময় লাগবে সেই বিষয়ে জানার একমাত্র উপায় হচ্ছে অ্যালগোরিদম এনালাইসিস। একটা সমস্যা অনেক ভাবে সমাধান করা যায়, কিন্তু আপনি কোন সমাধানটা বেছে নিবেন এটা আপনি বুঝবেন অ্যালগোরিদম এনালাইসিস করার

মাধ্যমে। এর পরে কিছু সার্চিং অ্যালগোরিদম যেমন বাইনারি সার্চ, লিনিয়ার সার্চ শিখে নিবেন। তারপরে শেখা যতে পারে কিছু সর্টিং অ্যালগোরিদম, যেমন - Bubble Sort, Insertion Sort, Selection Sort, Quick Sort এবং Merge Sort। আরও কিছু বিশেষ ধরনের অ্যালগোরিদম আছে যেগুলো খুব কম কিন্তু অনেক জটিল জটিল সমস্যা সহজে সমাধান করার কাজে ব্যবহৃত হয়। যেমন - Divide and Conquer, Dynamic Programming, Greedy Algorithm, Backtracking ইত্যাদি। এছাড়া প্রতিটা ডাটা স্ট্রাকচারের সাথেই আপনি অনেক গুলো করে অ্যালগোরিদম পাবেন। বিশেষ করে ট্রি এবং গ্রাফ এর সাথে তা অসংখ্য অ্যালগোরিদম রয়েছে।

## কমন অ্যালগোরিদমস

### Basic Algorithms

- Binary Search
- Bubble Sort
- Insertion Sort
- Merge Sort
- Quick Sort

### Advanced Algorithms

- Divide & Conquer
- Dynamic Programming
- Greedy Algorithm
- Back Tracking
- Graph Shortest Path Algorithms

ডাটা স্ট্রাকচার এবং অ্যালগোরিদম ভালোভাবে শেখার অর্থ হচ্ছে আপনি এখন কম্পিউটার জগতের স্থায়ী একজন বাসিন্দা। কেউ আপনাকে এই জগত থেকে বের করে দিতে পারবে না। আপনি নিজেই নিজের ক্যারিয়ার দাঢ় করাতে পারবেন কারোর সাহায্য ছাড়াই। এখন আপনি কম্পিউটারের সাথে কথা বলতে পারেন, তাকে বিভিন্ন ভাবে কমান্ড করতে পারেন, এমনকি এও জানেন যে কিভাবে কমান্ড করলে সব থেকে কম সময়ে এই বোকা মেশিনটা সব থেকে বেশি কাজ করবে। শুধু প্রোগ্রামিং ল্যাংগুয়েজ জানলে আপনি হয়ত কোড করতে পারবেন, ডাটা স্ট্রাকচার এবং অ্যালগোরিদম জানলে আপনার কোডের সৌন্দর্য রক্ষা হবে। এবং সব থেকে বড় বিষয়, কম্পিউটার সাইন্সের যে কোন শাখা প্রশাখা নিয়ে খেলা করার মত যোগ্যতা তৈরি হয়ে যাবে। এবার আপনি ডেভেলপমেন্ট করতে পারেন, ডাটা সাইন্স নিয়ে কাজ করতে পারেন, IoT, Robotics যে কোন কিছু নিয়েই কাজ করতে পারবেন। কারণ, আপনি এখন নিজের মনের মত করে বোকা মেশিনকে দিয়ে কাজ করিয়ে নিতে পারেন এবং সেটা এফিসিয়েন্ট ভাবে।

---

যদি আপনি প্রৱেশ সল্লিং, কম্পিউটিভ প্রোগ্রামিং এর কথা চিন্তা করে থাকেন তাহলে আপনার একমাত্র কাজই হবে ডাটা স্ট্রাকচার এবং অ্যালগোরিদম নিয়ে ধাঁটাধাঁটি করা। কারণ এখানে আপনার যে ধরনের প্রৱেশ সল্লি করতে হবে তার সব গুলোই কোনো না কোনো ডাটা স্ট্রাকচার এবং অ্যালগোরিদমের সাথে রিলেটেড। ডাটা স্ট্রাকচার এবং অ্যালগোরিদম যে শুধু আপনার কোডের সৌন্দর্যই বৃদ্ধি করবে এমনটা না। এটা আপনার কোডকে এফিসিয়েন্ট করবে, লক্ষ লক্ষ ডাটার জন্য ব্যবহারের উপযোগী করবে। আর সব থেকে বড় যেই কাজটি করবে সেটা হচ্ছে আপনার প্রৱেশ সল্লিং দক্ষতা বৃদ্ধি করবে। বাস্তব জীবনের যে কোনো সমস্যা সমাধান করার নতুন একটি দুয়ার আপনার সামনে উন্মুক্ত হয়ে যাবে।

ডাটা স্ট্রাকচার এবং প্রৱেশ সল্লিং এর জ্ঞান পরিপন্থ করতে আপনি ডিসক্রিট ম্যাথমেটিক্স (Discrete Mathematics) নিয়ে পড়াশোনা করতে পারেন। এটা ম্যাথমেটিক্সের একটি শাখা এবং এখানে ম্যাথম্যাটিক্যাল স্ট্রাকচার নিয়েই পড়াশোনা করা হয়। ম্যাথমেটিক্সের এই শাখাটি কম্পিউটার সাইন্সের সাথে সরাসরি সংযুক্ত এবং কম্পিউটার সাইন্সের শিক্ষার্থীদের বাধ্যতামূলক ভাবেই এই কোর্সটি গ্রহণ করতে হয়। আমি জানি, বেশির ভাগ শিক্ষার্থীই সেই সময় এই কোর্সটিকে কোনো ভাবে পাশ কাটিয়ে পার করে থাকে। কিন্তু বাস্তব জীবনে এই কোর্সের ভূমিকা অনেক। কারণ এই কোর্সের মুখ্য বিষয় গুলো হচ্ছে -

- Theoretical Computer Science
- Information Theory
- Logic
- Set Theory
- Combinatorics
- Graph Theory
- Probability
- Number Theory
- Algebraic Structures
- Calculus
- Geometry
- Topology
- Game Theory

- 
- Decision Theory
  - Discretisation

এই সব বড় বড় নাম দেখলে ভয় লেগে যাওয়া স্বাভাবিক। আর আমাদের কাছে তো ছোট বেলা থেকেই ম্যাথমেটিক্স একটা বিভীষিকার নাম। তবে সত্যি কথা বলতে ডিসক্রিট ম্যাথমেটিক্স এতটাও কঠিন কোনো বিষয় না। আপনাকে অনেকক্ষণ ধরে আমি ডাটা স্ট্রাকচার শিখতে বলছি, অ্যালগোরিদম শিখতে বলছি। ডিসক্রিট ম্যাথমেটিক্স হচ্ছে ডাটা স্ট্রাকচার গুলোর গাণিতিক মডেল। যদি ভালো ভাবে ডাটা স্ট্রাকচার আয়ত্ত করতে চান তাহলে সব সব থেকে সহজ সমাধান হবে এই ডিসক্রিট ম্যাথমেটিক্স। তাছাড়া বিভিন্ন অ্যালগোরিদমের কনসেপ্ট বুঝতে এবং নিজে নিজে অ্যালগোরিদম তৈরি করতে সাহায্য করবে ডিসক্রিট ম্যাথমেটিক্স। কম্পিউটার সাইন্সের যেকোনো ফিল্ডে কাজ করতে চাইলেই, এই ম্যাথমেটিক্সের জ্ঞান আপনার কাজে লাগবে। হোক সেটা ম্যাশিন লার্নিং বা হোক সেটা গেম ডেভেলপমেন্ট। আপনারা অনেকেই বলেন, ভাই লজিক আসে না। সিনট্যাক্স তো বুঝি, আপনি যখন করে দেখান তখন তো সহজই লাগে কিন্তু নিজে করতে গেলে পারি না। এই সমস্যার সমাধান আছে ডিসক্রিট ম্যাথমেটিক্স। যে কোনো একটা সাধারণ বিষয়কে কিভাবে লজিক্যাল ভাবে রিপ্রেসেন্ট করতে হয় তা শেখানো হয়েছে ম্যাথমেটিক্সের এই অন্যতম সুন্দর এই শাখাতে।

ডিসক্রিট ম্যাথমেটিক্স আয়ত্ত করতে খুব একটা সময়ও লাগবে না। ছোট বেলার মত শুধু পরীক্ষায় পাশের উদ্দেশ্যে যদি এটা শেখা শুরু করেন তাহলে সারা জীবনেও এটা শেখা সম্ভব না। শেখার সময় মাথায় রাখতে হবে, কম্পিউটারের জগতে সার্ভাইভ করার জন্য প্রোগ্রামিং ল্যাংগুয়েজ শেখা যেমন জরুরি ঠিক তেমনি নিজের লজিক বিউল্ড করাটাও জরুরি। আপনি হাজারটা প্রোগ্রামিং ল্যাংগুয়েজ জানলেও কেউ আপনাকে দাম দিবে না যদি না আপনি নিজের বুদ্ধিমত্তাকে কাজে লাগিয়ে বাস্তব সমস্যার সমাধান করতে পারেন। আর নিজের লজিক বিউল্ড করার স্ট্রাকচারড ওয়ে আমার কাছে মনে হয় ডিসক্রিট ম্যাথমেটিক্স শেখা। আপনাকে একবারেই সব কিছু শিখে ফেলতে হবে না। অল্প অল্প করে লজিক বিউল্ড করতে যেই টপিক্স গুলো কাজে লাগবে সেগুলো দিয়ে শুরু করুন। সপ্তাহে ৩-৪ ঘণ্টা সময় দিয়ে একটা টপিক্স আয়ত্ত করুন এবং এর বাস্তব প্রয়োগ দেখুন। আমার বিশ্বাস, কিছু দিন পরে আর আপনাকে বলতে হবে না যে ডিসক্রিট ম্যাথমেটিক্স শিখুন। আপনি নিজেই এর প্রয়োজনীয়তা উপলব্ধি করতে পারবেন।

আমি নিজেও আপনাদের অনেকের মতই ফাঁকিবাজ ছিলাম। আর স্কুল জীবন থেকেই প্রোগ্রামিং এর সাথে নিজেকে যুক্ত করায় একাডেমিক পড়াশোনাতে অনেক ফাঁকি দিয়েছিলাম। পরবর্তীতে অবশ্য যখন বুঝতে পেরেছিলাম গণিত ছাড়া গতি নেই, তখন আবার ক্লাস থ্রি থেকে টেন পর্যন্ত গণিত বই একবার ঝালায় করে নিয়েছিলাম। আর সব থেকে বেশি মজা পেয়েছিলাম ডিসক্রিট ম্যাথমেটিক্স করতে গিয়ে। আগে থেকেই প্রোগ্রামিং এর কিছু জ্ঞান থাকায় ডিসক্রিট ম্যাথমেটিক্সের ব্যবহার সম্পর্কে বুঝতে পারছিলাম। জীবনে প্রথমবারের মত ম্যাথমেটিক্সের বাস্তব প্রয়োগ গুলো চাখের সামনে ভাসছিল। বুঝতে পারছিলাম কেন এত দিন ছোট ছোট প্রোগ্রামিং প্রলৈম সল্লু করতে গেলে নিজের মাথার চুল ছিঁড়তে হতো। খুব আফসোসও হচ্ছিল যে কেন আরও আগে আমি ম্যাথমেটিক্সের এই শাখাটা সম্পর্কে জানতে পারলাম না।

আমি নিজে খুব ভালো ম্যাথমেটিক্স পারি না। তারপরেও ডিসক্রিট ম্যাথমেটিক্সের বেশির ভাগ বিষয় বুঝতে আমার কোনো সমস্যায় হ্যনি। অবশ্য শেখার সময় আমি জানতাম কেন শিখছি, এও জানতাম যে আমাকে শিখতেই হবে। এই জন্য হতে পারে খুব একটা সমস্যার সম্মুখীন হ্যনি। তবে আমি যেহেতু পেরেছি, আমার বিশ্বাস আপনিও পারবেন। আর যারা প্রথম ম্যাথমেটিক্সের এই শাখা সম্পর্কে শুনলেন তারা তাদের লজিক বিউল্ড করার ক্ষেত্রে যদি ডিসক্রিট ম্যাথমেটিক্স থেকে যদি কোনো উপকার পেয়ে থাকেন তাহলে মনে মনে আমাকে ধন্যবাদ দিতে ভুলবেন না।

## ডিসক্রিট ম্যাথমেটিক্সের বিষয় সমূহ

### Basic Concepts

- Logic & Proofs
- Basic Structures
- Fundamental Algorithms
- Integers and Matrices
- Induction & Recursion
- Clouting Techniques
- Discrete Probability

### Advanced Concepts

- Relations
- Graps
- Trees
- Boolean Algebra
- Calculas
- Topology
- Decision Theory

---

প্রথম দিকে ডাটা স্ট্রাকচারের শুরুত্ব খুব একটা উপলক্ষ্মি করা যায় না। এমনকি ডেভেলপমেন্টে গেলেও আপনি সচরাচর এর ব্যবহার দখতে পারবেন না। তার কারণ হচ্ছে আমরা যেই ধরনের অ্যাপ্লিকেশন তৈরি করে থাকি তা খুব সীমিত ইউজার ব্যবহার করে থাকে এবং সেখানে ডাটাও অনেক কম থাকে। তাই আমরা যদি ইনেফিসিয়েন্ট ভাবেও কোড করি, কোনো দৃশ্যমান পরিবর্তন লক্ষ্যনীয় হয় না। কিন্তু এই ইনেফিসিয়েন্ট কোড যখন কোটি কোটি ডাটার জন্য ব্যবহৃত হবে, তখন অ্যাপ্লিকেশনটা আর ব্যবহারের উপযোগী থাকবে না। ফেসবুক তার একটা বড় উদাহরণ। কিছুক্ষণ ফেসবুক ব্রাউজ করার পরেই হ্যাঁ হওয়া শুরু হয়। কারণ এখানে প্রতি মুভর্টে নতুন নতুন ডাটা আসতে থাকে, হাজার হাজার ক্যাল্কুলেশন হতে থাকে। ফেসবুক কোটি কোটি টাকা খরচ করে, বিশ্বের সব থেকে বড় বড় ডেভেলপারদের দিয়ে এফিসিয়েন্ট ভাবে কোড লেখার পরেও তারা সম্পূর্ণ পার্ফর্মেন্স এর গ্যারান্টি দিতে পারছে না। তাহলে একটা ইনেফিসিয়েন্ট কোডের কি হবে আপনারাই বিবেচনা করুন।

বড় বড় কোম্পানিতে আপনি যখন জবের জন্য ইন্টার্ভিউ দিবেন, আপনি কয়টা প্রোগ্রামিং ল্যাংগুয়েজ জানেন, কয়টা ফ্রেমওয়ার্ক জানেন এই গুলো তারা জিজ্ঞাসা করবে না। তাদের ইন্টার্ভিউ এর মুখ্য বিষয়ই থাকবে আপনি কম্পিউটার সাইন্স কর্তৃ জানেন, কত গুলো ডাটা স্ট্রাকচার এবং অ্যালগোরিদম সম্পর্কে আপনার জ্ঞান আছে। বাস্তব কোনো সমস্যা আপনার সামনে আসলে ঠিক কোন ডাটা স্ট্রাকচারটা ব্যবহার করে আপনি সমস্যাটার সমাধান করছেন, কেন সেই ডাটা স্ট্রাকচারটি বেছে নিলেন আর আপনার করা সমাধান কর্তৃ এফিসিয়েন্ট মানে কত ভালো ভাবে আপনি অ্যালগোরিদমটি ডিজাইন করলেন। এই গুলোই তাদের জানার বিষয় থাকে। আর প্রশ্ন গুলোর সঠিক উত্তর যদি আপনার জানা থাকে তাহলে যে কোনো ল্যাংগুয়েজ, যে কোনো ফ্রেমওয়ার্ক নিয়ে কাজ করা আপনার জন্য কোনো মুখ্য বিষয় না। তাহলে নিশ্চয় বোঝা যাচ্ছে যে, ডাটা স্ট্রাকচার এবং অ্যালগোরিদম কর্তৃ শুরুত্বপূর্ণ একটা বিষয়?

আর এই শুরুত্ব পূর্ণ বিষয়টি শেখার জন্য আপনাকে যথেষ্ট পরিমাণ সময় ব্যয় করতে হবে। ডাটা স্ট্রাকচারের থিওরি ভালো ভাবে বুঝতে হবে। কোথায় কখন এবং কেন কোন ডাটা স্ট্রাকচার বা অ্যালগোরিদমটি ব্যবহার করবেন সেটা ভালো ভাবে বুঝতে হবে। ডাটা স্ট্রাকচারও অবজেক্ট অরিয়েন্টেডের মতই একটা ইউনিভার্সাল থিওরি, কোনো ল্যাংগুয়েজের সাথে সম্পৃক্ত বিষয়বস্তু না। তাই থিওরি ভালো ভাবে আয়ত্ত করে কয়েকটা ল্যাংগুয়েজে ইমপ্লিমেন্ট করার চেষ্টা করতে হবে। মনে রাখবেন, প্রোগ্রামিং ল্যাংগুয়েজ শেখা যতটা শুরুত্বপূর্ণ, একজন

ভালো প্রোগ্রামার হিসেবে নিজেকে গড়ে তোলার জন্য ডাটা স্ট্রাকচার এবং অ্যালগোরিদমের জ্ঞানও ঠিক ততটাই শুরুত্বপূর্ণ।

## ডাটা স্ট্রাকচার ইমপ্লিমেন্ট করবো কোন ল্যাংগুয়েজে

**সিঃ** ডাটা স্ট্রাকচার এবং অ্যালগোরিদম ইমপ্লিমেন্টেশন শখার অন্যতম একটা ল্যাংগুয়েজ হচ্ছে সি। আপনি যদি প্রৱেশ সন্তুষ্ট হয়ে থাকেন তাহলে বেশির ভাগ ক্ষেত্রেই আপনি সি অথবা সি++ ব্যবহার করেই ডাটা স্ট্রাকচারস ইমপ্লিমেন্ট করবেন। তবে আমার মনে হয়, এত লো লেভেল ল্যাংগুয়েজ ব্যবহার করে ডাটা স্ট্রাকচার ইমপ্লিমেন্ট করা শিখতে চাইলে আমাদের অনেক বেশি কষ্ট করতে হবে। আমাদের ফোকাসটা ডাটা স্ট্রাকচারের থিওরি থেকে নড়ে গিয়ে ল্যাংগুয়েজ এর ফিচারের ওপরে চলে যাবে। যখন ট্রি গ্রাফ এর মত ডাটা স্ট্রাকচার ইমপ্লিমেন্ট করতে হবে তখন প্রচুর কোড লিখতে হবে। এই ক্ষেত্রে সি++ যদিও একটা ভালো সমাধান হতে পারে। তবে আপনি যদি ল্যাংগুয়েজ এবং ডাটা স্ট্রাকচার একই সাথে বুঝতে চান, সেই ক্ষেত্রে আপনি সি ল্যাংগুয়েজ ব্যবহার করতে পারেন।

**জাভা:** আমি পার্সোনালি মনে করি ডাটা স্ট্রাকচার এবং অ্যালগোরিদম জাভা ব্যবহার করেই ইমপ্লিমেন্ট করা উচিত। বাজারে আপনি প্রচুর বইও পাবেন এই বিষয়ের ওপরে যা একই সাথে আপনার অবজেক্ট অরিয়েন্টেড থিওরি এবং ডাটা স্ট্রাকচারের জ্ঞান সমৃদ্ধ করবে। জাভা যেহেতু হাই লেভেল ল্যাংগুয়েজ তাই পুরো ফোকাসটা আপনি ডাটা স্ট্রাকচার এবং এর থিওরি এর ওপরেই দিতে পারবেন। জাভাতে কোড অর্গেনাইজ করাটাও অনেক সহজ হওয়ায় যেকোনো ধরনের ডাটা স্ট্রাকচার খুব সহজেই জাভাতে ইমপ্লিমেন্ট করা যায়।

**পাইথন:** যদি আপনার টাগেটি থাকে শুধুমাত্র ডাটা স্ট্রাকচারের ওপরেই ফোকাস করবেন এবং ইমপ্লিমেন্ট করে শিখবেন তাহলে পাইথন বেষ্ট হবে। কারণ এখানে আপনাকে কোডিং নিয়ে মোটেও চিন্তা করতে হবে না। অল্প কয়েক লাইন কোড লিখেই পাইথনে অনেক কিছু করা যায়। ডাটা স্ট্রাকচার এবং অ্যালগোরিদম ইমপ্লিমেন্ট করার অন্যতম একটা সমাধান হচ্ছে পাইথন। জাভার মত পাইথনেও আপনি অসংখ্য ভালো ভালো বই পাবেন।

## ডাটা স্ট্রাকচার এবং অ্যালগোরিদম শেখার রেফারেন্স

### ডাটা স্ট্রাকচার এবং অ্যালগোরিদমের বই সমূহঃ

- Introduction to Algorithm By Thomas H. Cormen
- The Algorithm Design Manual By Steven Skiena
- Algorithms By Robert Sedgewick
- Data Structures and Algorithm in Java By Michael T. Goodrich
- Introduction to the Design and Analysis of Algorithms By Anany Levitin

### ডাটা স্ট্রাকচার এবং অ্যালগোরিদমের ইউটিউব চ্যানেলঃ

- [Tamim Shahriar](#) (Bangla)
- [LoveExtendsCode](#) (Bangla)
- [Stack Learner](#) (Bangla) [Upcoming]
- [MIT Open Course](#) (English)
- [Applied Course](#) (English)
- [Jenny's Lectures](#) (English)

### ডাটা স্ট্রাকচার এবং অ্যালগোরিদমের ওয়েবসাইটঃ

- [Geeks For Geeks](#)
- [Programmiz](#)
- [JavaTPoint](#)

### ডাটা স্ট্রাকচার এবং অ্যালগোরিদম ট্রেনিং প্রোগ্রাম এবং কোর্সেসঃ

- [Stack Learner Premium Courses](#) (Upcoming)
- [Stack Learner Bootcamps](#) (Training Program)

বিঃদ্রঃ একটি নির্দিষ্ট সময়ে একটি নির্দিষ্ট রিসোর্স মনে চলবেন

## ডিসক্রিট ম্যাথেমেটিক্স শখার রেফারেন্স

### ডিসক্রিট ম্যাথেমেটিক্সের বই সমূহঃ

- Discrete Mathematics And It's Application By Kenneth H. Rosen
- Discrete Mathematics: An Open Introduction By Oscar Levin
- Concrete Mathematics By Donald Knuth

### ডিসক্রিট ম্যাথেমেটিক্সের ইউটিউব চ্যানেলঃ

- দ্বিমিক কম্পিউটিং (Bangla)
- Anisul Islam (Bangla)
- Judemy(Bangla)
- Stack Learner (Bangla) [Upcoming]
- Naso Academy (English)
- The Trev Tutor (English)
- Trefor Bazett (English)

### ডিসক্রিট ম্যাথেমেটিক্স ওয়েবসাইটঃ

- Brilliant
- Discrete Mathematics - An Open Introduction
- Journals Elsevier

বিঃদ্রঃ একটি নির্দিষ্ট সময়ে একটি নির্দিষ্ট রিসোর্স মনে চলবেন



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>

# ৮

---

অধ্যায় চার  
সাহিত্য যাত্রা

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ ডেভেলপমেন্ট কি
- ✓ ডেভেলপমেন্ট প্রসেস
- ✓ ওয়েব ডেভেলপমেন্ট
- ✓ কমন ডেভেলপমেন্ট টুলস

---

ডেভেলপমেন্ট - কত রং বেরং এর বাহারি রকমের ডেলপমেন্ট যে আছে দুনিয়াতে তার ইয়ত্তা নেই। গেম ডেভেলপমেন্ট, আপ ডেভেলপমেন্ট, ওয়েব ডেভেলপমেন্ট। পরিচিত এই ডেভেলপমেন্ট গুলোর নাম শুনেই, ডেভেলপারদের লাইফ স্টাইল, স্যালারি দূর থেকে দেখেই তো আমাদের মনে ডেভেলপার হওয়ার স্পষ্ট প্রথম উँকি দেয়। যখন আমি কম্পিউটারও ভালো ভাবে ব্যবহার করতে জানতাম না, তখন একটা কথা কিন্তু খুব ভালো ভাবেই জানা হয়ে গিয়েছিল আমার। কোনো ভাবে একবার ভাল একজন ডেভেলপার হতে পারলেই জীবনে আর কোনো কিছুর চাহিদা থাকবে না। ঘরে বসেই বসেই তখন ডলার ছাপাতে পারবো। কথাটা একটু বেশি বেশি মনে হলেও কথা কিন্তু সত্য। ভাল একজন ডেভেলপার অন্য আর দশটা ভালো পেশার মানুষের থেকে কিন্তু বেশিই উপার্জন করতে পারে ঘরে বসে থেকেই। তবে ভাল মাপের ডেভেলপার হতে হবে।

আমরা শুরুর দিকে যেই ভুলটা প্রথমেই করে বসি সেটা হল, শুধু ডেভেলপার হওয়ার কথা চিন্তা করি। ভালো মাপের ডেভেলপার যে হতে হবে সেই কথাটা অনেকেই ভুলে যায়। ঘরে বসে বসে বিদেশি মুদ্রা উপার্জন করার আর বিলাসী জীবন যাপন করার লোভ আমাদের পেয়ে বসে। যেই কারণে আমরা খুব দ্রুতই ডেভেলপার হওয়ার জন্য বিভিন্ন শর্টকাট রাস্তা খোঁজার চেষ্টা করি। কিন্তু একটা কথা মনে রাখতে হবে আপনাকে। ডেভেলপমেন্টের যাত্রা, বিশেষ করে শুরুর দিকের যাত্রা খুবই কষ্টদায়ক। এখানে আপনাকে বিশেষ ধরনের ধৈর্যের পরীক্ষা দিতে হবে। শুধু দূর থেকেই মনে হবে যে ঘরে বসে বসে কম্পিউটারের কীবোর্ড নিয়ে গুঁতাগুঁতি করা, এ আর এমন কি কঠিন কাজ। কম্পিউটারের সামনে বসে থেকেই যদি উপার্জন করা যায় তাহলে নাহয় কয়েক ঘণ্টা বেশিই বসে থাকলাম। আসলে ডেভেলপাররা কম্পিউটারের সামনে বসে থাকার জন্য অর্থ পায় না, অর্থ পায় বাকি আর দশটা পেশার মতই নিজের দক্ষতা, বুদ্ধিমত্তা বিক্রি করার জন্য।

আমাদের দেশে সরাসরি ডেভেলপমেন্টে জাম্প করার ব্যাপারটা খুবই সাধারণ। প্রোগ্রামিং, ডাটা স্ট্রাকচার, অ্যালগোরিদম বা কম্পিউটার সাইন্সের সাধারণ বিষয় গুলোও যে গুরুত্বপূর্ণ, সেই বিষয়টা মানুষের জানার বাইরেই থেকে

যায়। যার ফলে কি হয়, কিছু দূর যাবার পরে আর কুল কিনারা খুঁজে পাওয়া যায় না। ডেভেলপমেন্ট হচ্ছে ফাইনাল প্রোডাক্ট। আর কোনো বিষয়ে সঠিক জ্ঞান না থাকলে ভালোভাবে একটা প্রোডাক্ট তৈরি করা যায় না। একটা উদাহরণের মাধ্যমে ব্যাপারটা বোঝার চেষ্টা করা যাক।

আপনি যদি একটা গাড়ি বানাতে চান তাহলে আপনার কি দরকার হবে? অবশ্যই গাড়ি যেই সূত্র গুলো মেনে রাস্তায় চলবে সেই সূত্র গুলো জানার দরকার পরবে, কিছু ইঞ্জিনিয়ারিং জ্ঞানের দরকার পরবে। গতির সূত্র, ঘর্ষণের সূত্র, চাপের সূত্র না জেনেই কি আপনি গাড়ি বানাতে পারবেন? সূত্র আপনার আবিষ্কার করার দরকার নেই। সেটা নিউটন সাহেব, প্যাসকেল সাহেব অনেক আগেই করে রেখে গেছেন। আপনার কাজ সূত্র গুলোকে এনালাইসিস করে আপনি যে নতুন গাড়িটি বানাবেন তার সমস্যা গুলো দূর করা এবং এমন একটি গাড়ি বানানো যা দেখে ব্যবহারকারীরা রীতিমত বিস্মিত হয়ে যাবে। কিন্তু আপনি কি করলেন, বললেন এই সব সূত্র জেনে কি হবে? বাজারে অনেক গাড়ি আছে, নিয়ে এসে খুলে দেখব এবং কিছু পার্টসম্পত্র পরিবর্তন করে নিজের নামে চালাবো। এবার বলুন, যদি আপনি এই কাজই করেন তাহলে আপনাকে কি নামে ডাকা উচিত? ইঞ্জিনিয়ার নাকি মেকানিক?



### What is Software Development?

- Software development is the process of conceiving, specifying, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining applications, frameworks, or other software components. ([Wikipedia](#))

দুঃখজনক হলেও সত্যি আমাদের দেশে ডেভেলপমেন্ট সেক্টরটা বেশিরভাগ ক্ষেত্রেই এরকম ভাবে চলছে। এর বাইরেও কিছু মানুষ আছে, যারা নিজের জ্ঞানকে সঠিক ভাবে বৃদ্ধি করছে। আর তাদেরকে তুলে নিয়ে যাচ্ছে, গুগল মাইক্রোসফট এর মত কোম্পানিরা, আমাদের দেশে গুগল মাইক্রোসফট তৈরি হওয়ার স্বপ্নটাকে ধূলিসাং

---

করে দিয়ে। যেখানে অন্ন বিদ্যার কারণে এবং দ্রুত ইনকামের আশায় আমরা শ্রম বিক্রি করছি বিভিন্ন অনলাইন প্লাটফর্মে সেখানে একটু শ্রম দিয়ে নিজেকে দক্ষ করলে হয়ত সার্ভিস বিক্রি করতে পারতাম। তখন হয়ত আমাদের দেশ দশগুণ বেশি বৈদেশিক মুদ্রা অর্জন করতে পারত। আমরা নিজেরাও হয়ত অনেক বেশি লাভবান হতাম।

একটা বিষয় হয়ত আপনি লক্ষ্য করবেন, যখন কোনো শিক্ষার্থী মেডিক্যালে ভর্তি হয়, তখন ৫ বছর টানা তাকে পরিশ্রম করতে হয় তার পড়াশোনা নিয়ে। এর মাঝে কোন বিরতি নেই। বাড়তি উপর্যুক্তির কোনো উপায় নেই। কোনো শিক্ষার্থীর স্বাস্থ্য যদি একটু ভালো থাকে, ডাক্তার হয়ে বের হতে হতে তার চেহারা পুরো পাল্টে যায়। নিজে ডাক্তার হলেও দেখতে কয়েক বছর রোগীর মতই লাগে। ডাক্তারি পাশ করে বের হওয়ার পরে দুই তিন বছর বিভিন্ন হাসপাতালে অনুশীলন করার পূর্ব মূল্য পর্যন্ত আমরা তাকে ডাক্তার বলতে নারাজ। এত বছর, এত গাধার মত পরিশ্রম করার পরেও আমরা তাকে ডাক্তারের মর্যাদা দিতে চাই না।

আবার ধরেন একজন আর্কিটেক্ট। ৫ বছরের কোর্স পুরোপুরি ভাবে সম্পূর্ণ করার পরে, দুই একটা ফার্মে ইন্টার্ন করার পরেও আপনি তার হাতে আপনার বাড়ির ডিজাইন করতে দিবেন না। কেন? আপনার মনে হবে যে সে এখনো অভিজ্ঞ না। একজন উকিলের কথায় চিন্তা করেন না, আমার নিজের খালু আজ প্রায় ৭ বছর হাই কোর্টে উকালতির প্রাকটিস করছেন। এখনো তিনি জুনিয়রই রয়ে গেছেন। হাই কোর্টে যাওয়ার পূর্বে কিন্তু বহু বছর তিনি নিজ এলাকাতেও উকালতি করেছেন।

এত এত উদাহরণ দিয়ে আমি আসলে কি বোঝাতে চাচ্ছি? বোঝাতে চাচ্ছি কম্পিউটার সাইন্সের গুরুত্ব। আমার মনে হয় আমাদের দেশে কম্পিউটার সাইন্সের কোন গুরুত্ব নেই। একজন সদ্য কম্পিউটার সাইন্সে ভর্তি হওয়া শিক্ষার্থী ফ্রিলান্সিং প্লাটফর্মে অ্যাপ্লিকেশন ডেভেলপমেন্টের কাজের জন্য অ্যাপ্লাই করে। অ্যাপ্লাই করাটা দোষের কিছু না, কারণ তার আগে থেকেই সেই বিষয়ে দক্ষতা থাকতে পারে এবং এই সাইট গুলোতে সাধারণত খুব ছোট ছোট কাজই পাওয়া

---

যায়। কোনো বড় কোম্পানি তাদের বড় কোনো প্রজেক্ট ফ্রিলান্সিং সাইট গুলোতে দেয় না। আপনি ফ্রিলান্সার হতে চাচ্ছেন এটা খুবই ভালো উদ্যোগ। আপনি নিজের কর্ম সংস্থান নিজেই করবেন আমাদের দেশের প্রেক্ষাপটে এর থেকে ভাল খবর আর কি হতে পারে? তবে চিন্তা করুন তো, আপনি কোন সময়ে ফ্রিলান্সিং করছেন? আপনার জীবনের গোল্ডেন মুহূর্তে নয় তো? যেই সময়টা আপনি ব্যয় করছেন কোন একজন ক্লাইণ্টের একটা পার্সোনাল সাইট বানানোর জন্য, সেই সময়টাকে কাজে লাগিয়ে সঠিক জ্ঞান অর্জন করলে হয়ত একদিন আপনি এমন কোনো সার্ভিস তৈরি করতে পারতেন যা বিশ্বে লক্ষ লক্ষ মানুষ আপনার কাছ থেকে বৈদেশিক মুদ্রার বিনিময়ে কিনতো।

আসলে সব কিছুর একটা সময় আছে, ফ্রিলান্সিং একটা ভাল পেশা। কিন্তু এই পেশায় সঠিক ভাবে টিকে থাকতে চাইলে দরকার দক্ষতা। আর এই দক্ষতা অর্জন করতে দরকার সময়। ভাবতে খুব খারাপ লাগে আমাদের দেশের কিশোর কিশোরীদের ভিতরে এই ফ্রিলান্সিং এর ভূতটা চুকে গেছে। তারা মনে করে কম্পিউটারের সামনে বসলেই অনেক টাকা। ফ্রিলান্সিং তাদের কম্পিউটারের প্রতি আগ্রহ তো জাগ্রত করেছে, কিন্তু বেশির ভাগ ক্ষেত্রেই তাদের ভুল পথে ধাবিত করেছে। অনেকে তাদের এই মোহকে কাজে লাগিয়ে অনেক অসৎ কাজ করিয়ে নিচ্ছে অল্প কিছু টাকার বিনিময়ে।

নিজের উপার্জন করা প্রথম অর্থটা সবার কাছেই গুরুত্বপূর্ণ। তাই অনেকেই অনলাইন থেকে ২০-২৫ ডলার উপার্জন করার পরেই সেটা সবার সাথে শেয়ার করে, খুব খুশি হয়। খুশি হওয়া বা শেয়ার করা কোনো দোষের কিছু না। তবে আপনাকে জানতে হবে যে আপনি কি বিক্রি করে এই অর্থ উপার্জন করছেন। একটু খোঁজ নিয়ে দেখলেই দেখা যায়, যে এই ২০-২৫ ডলার উপার্জনের জন্য তারা তাদের ৩-৪ মাস সময় নষ্ট করেছে, পড়াশোনা শিকায় তুলে রেখেছে। শুনতে খারাপ শোনালেও একটা উদাহরণ দেই, প্রতিদিন সকালে ৮ বছরের যেই ছেলেটা ঝিগাতলা টু ফার্মগেট রোডে লেগুনার হেল্পারি করে সেও দিন শেষে ৪০০ টাকা

---

বাসায় নিয়ে যায়। মানে মাসে ১২০০০ টাকা। তাহলে আপনি যদি ৪ মাসে ২০ ডলার বা ১০০ ডলারই উপার্জন করেন তাহলে আপনার মাসিক ইনকাম কত? কম্পিউটার, মডেম, ইন্টার্নেট বিল দিয়ে আপনার ইনভেস্টমেন্টই বা কত আর প্রতি মাসে খরচই বা কত? ইনকাম কত সেটা এখানে মুখ্য বিষয় না, মুখ্য বিষয় হচ্ছে ওই লেগুনার হেল্পার আগামীকালকে বিসর্জন দিয়ে আজকে উপার্জন করছে, আপনিও কি তাই করছেন না? আপনিও কি আপনার ভবিষ্যৎ বিক্রি করে বর্তমানে অন্ন কিছু টাকা উপার্জন করছেন না? আপনার ওপরে যদি পরিবারের দায়িত্ব না থাকে তাহলে নিজেকে দক্ষ করে গড়ে তুলুন, আর দক্ষ হওয়ার পরেই উপার্জনের চেষ্টা করুন।

ডেভেলপমেন্ট কোনো ছেলেখেলার বিষয় নয়। এর জন্য দরকার সঠিক দিকনির্দেশনা এবং অধ্যাবসায়। প্রোগ্রামিং হচ্ছে ভাষা, ডাটা স্ট্রাকচার এবং অ্যালগোরিদম হচ্ছে ভাষার ব্যাকরণ আর ডেভেলপমেন্ট হচ্ছে এই ভাষায় রচিত সাহিত্য। ভাষা এবং ব্যাকরণ না জেনে যদি আপনি সাহিত্য লিখতে বসে পড়েন, তাহলে লেখা তো হবে, সেটা সাহিত্য হবে কিনা তা নিয়ে ভাবনা চিন্তার বিষয় আছে। অনেক বড় বড় মানুষ, বড় কোম্পানির চটকদার সব বিজ্ঞাপন দেখে আপনি যদি ভাবেন যে ২-৩ মাসেই আপনি বড় ডেভেলপার হয়ে যাবেন, তাহলে আপনি এখনো স্বপ্নের রাজ্যে বাস করছেন। এবার সময় হয়েছে চোখটা মেলে বাস্তবতার সামনা সামনি হওয়ার। একটু সহজ ভাবে ভাবলেই ব্যাপারটা আপনার কাছে পরিষ্কার হয়ে যাবে। ছেট বেলা থেকেই আমরা সবাই ক্রিকেট খেলি, কিন্তু একজন ভালো ক্রিকেটার হওয়ার জন্য কত বছরের অধ্যাবসায় দরকার? ডেভেলপমেন্ট কি ক্রিকেট খেলার থেকেও সহজ?

আপনারা হয়ত ভাবছেন, আমি আপনাদেরকে ভয় দেখাচ্ছি। না, আমি ভয় দেখাচ্ছি না। আমি শুধু আর সবার মত মিথ্যা আশ্বাস দিচ্ছি না, বাস্তবতাটা তুলে ধরার চেষ্টা করছি। বল মানুষ যুগ যুগ ধরে আপনাদেরকে মোটিভেট করে আসছে, আর মোটিভেট হতে হতেই দেশের সফটওয়্যার ইন্ডাস্ট্রির আজকের এই দশা। তারা মোটিভেট করেছে যেন মানুষ এই ইন্ডাস্ট্রি এর দিকে এগিয়ে আসে, অনেক

---

বেশি মানুষ কাজ শিখে দেশের মুখ উজ্জ্বল করে। আর মানুষ বুঝেছে এখানে গেলেই টাকা। এখন মোটিভেশনের সময় না, বাস্তবতাকে মেনে নিয়ে সামনে এগিয়ে যাওয়ার সময়। কেউ যখন মিলিটারিতে নিজেকে যুক্ত করে, এটা জেনে নিয়েই নিজেকে যুক্ত করে যে দেশের জন্য যে কোন মুভর্তে তাকে প্রাণ দিতে হতে পারে। তাই আপনাকেও প্রথমে জানতে হবে যে ডেভেলপার হতে হলে আপনাকে পরিশ্রম করতেই হবে, শুধু শুধু ঘরে বসে থাকলে কেউ এসে টাকা দিয়ে যাবে না। এখানে পরিশ্রমের মাত্রাটাও অন্য যেকোনো কার্যক শ্রমের থেকেও অনেক বেশি। অনেক সময় হাতে নিয়ে বসতে হবে, অনেক ধৈর্য বুকে নিয়ে বসতে হবে, অনেক ত্যাগ স্বীকার করার মানুষিতা নিয়ে বসতে হবে। একটা দেশে যেমন প্রয়োজনের সময় জীবন দেওয়ার মত সৈনিক দরকার, ঠিক তেমনি আপনার মত জ্ঞান গর্ভ সফটওয়্যার ডেভেলপারও দরকার দেশের সফটওয়্যার চাহিদা মিটিয়ে বাইরে রপ্তানি করে দেশের মুখ উজ্জ্বল করার জন্য।

সব থেকে মজার বিষয় কি জানেন, আপনি পারবেন। আপনি কোথায় পড়াশোনা করছেন, আপনি সাইন্সের স্টুডেন্ট কিনা, আপনি কম্পিউটার সাইন্স পড়েন কিনা, কোন কিছুই আসলে আপনার পথের কাঁটা হতে পারবে না যদি আপনি নিজ উদ্দ্যোগে, স্বশিক্ষায় শিক্ষিত হয়ে দেশের একজন সনামধন্য সফটওয়্যার সৈনিক হিসেবে নিজেকে তৈরি করতে চান। আর তার জন্য আপনাকে কি করতে হবে, কি শিখতে হবে তার অনেক কিছু আমি এই বই এর মাধ্যমে জানানোর চেষ্টা করবো। অন্ততপক্ষে একটা সুন্দর যাত্রার সাক্ষী হওয়ার চেষ্টা করবো।

পৃথিবীতে অনেক বকমের ডেভেলপমেন্ট আছে। আপনার যদি প্রোগ্রামিং এর জ্ঞান পরিষ্কার থাকে তাহলে আপনি যেকোনো ফিল্ডেই অনেক ভাল করতে পারবেন। তবে আমি নিজে যেহেতু ওয়েব ডেভেলপার, আর এই বইটাও ওয়েব ডেভেলপারদের জন্যই লেখা তাই আমি এখানে শুধু ওয়েব ডেভেলপমেন্ট বিষয়েই আলোচনা করবো। ওয়েব ডেভেলপমেন্ট, একটা বিরাট বড় জার্নি। এই জার্নিতে আপনাকে একা সাফার করতে হবে না। অনেক মানুষ মিলে জার্নিটাকে সফল

---

করে। অনেক গুলো সেক্টর ওয়েব ডেভেলপমেন্টের সাথে জড়িত। আপনি ইচ্ছে করলে যে কোন একটি সেক্টর বেছে নিতে পারেন, আবার ইচ্ছে করলেই সব গুলো সেক্টরের সাথেই নিজেকে জড়াতে পারেন। তবে একটা একটা করে সেক্টরে নিজের পরিধি বিস্তার করাটাই সব থেকে বৃদ্ধিমানের কাজ হবে।

একটা অ্যাপ্লিকেশন সেটা যে কোন ধরনের অ্যাপ্লিকেশনই হতে পারে, আইডিয়া থেকে প্রোডাকশনে যাওয়া পর্যন্ত মানে ইউজারের হাতে ব্যবহারের জন্য যাওয়া পর্যন্ত অনেকগুলো ধাপ পার করতে হয়। অনেক চড়াই উৎরায়ের পরেই একটা অ্যাপ্লিকেশন জীবনের মুখ দেখে। একজন ডেভেলপার হিসেবে আমাদের উচিং শুরু থেকে শেষ পর্যন্ত সম্পূর্ণ প্রসেসটা সম্পর্কে নূন্যতম হলেও একটা আইডিয়া রাখা। এখানে আমি কিন্তু সাধারণ ওয়েব সাইটের কথা বলছি না, বলছি অ্যাপ্লিকেশনের কথা। আর একটা অ্যাপ্লিকেশন কিভাবে ডেভেলপড হয়, সেটা যদি আগে থেকেই জানা থাকে তাহলে নিজের অবস্থান পরিষ্কার করতে সহজ হয়। নিজেকে যে কোনো এক বা একাধিক জায়গার জন্য আগে থেকেই প্রস্তুত করে নেওয়া যায়। সফটওয়্যার ডেভেলপমেন্ট প্রোসেসটা যে কোনো প্রোডাক্ট তৈরি করার থেকে কম কিছু না। এখানেও অসংখ্য মানুষ কাজ করে ভিন্ন ভিন্ন ধরনের অপারেশন সম্পর্ক করার জন্য। একটা প্রোডাক্ট তৈরি করা আর সফটওয়্যার ডেভেলপ করার মধ্যে মূল পার্থক্য হল, একটা প্রোডাক্ট ভিন্ন ভিন্ন কাঁচামাল থাকে আর এখানে আপনি, আপনার দক্ষতায় হচ্ছে কাঁচামাল। আর আপনার কম্পিউটারটা হচ্ছে প্রোডাকশন মেশিন।

একটা আইডিয়া যখন গ্রহণযোগ্যতা পায়, তখন প্রথম কাজ হচ্ছে এই আইডিয়াটাকে বাস্তব রূপ দেওয়ার জন্য প্রয়োজনীয় রিকুয়ারমেন্ট কালেক্ট করা। সাধারণত সমস্ত আইডিয়াই খুব ছোট এবং সিম্পল থাকে। এই আইডিয়াকে বাস্তবায়ন করতে, মানুষের ব্যবহারের উপযোগী করে গড়ে তুলতে কি কি বিষয় মাথায় রাখতে হবে, কি কি ফিচার ইমপ্লিমেন্ট করতে হবে এই সব খুঁজে বের করাই রিকুয়ারমেন্ট ইঞ্জিনিয়ারের কাজ। তারা অনেক দিন ধরে রিসার্স করে একটা স্পেসিফিকেশন তৈরি করে যাকে বলা হয় Software Requirement

---

**Specification.** রিকুয়ারমেন্ট খুব গুরুত্বপূর্ণ বিষয়, কারণ প্রথম যখন অ্যাপলিকেশনটার আর্কিটেকচার তৈরি করা হয় তখন এই ইনিশিয়াল রিকুয়ারমেন্টের ওপরে ভিত্তি করেই তৈরি করা হয়। তবে পরবর্তীতে রিকুয়ারমেন্ট পরিবর্তন বা পরিবর্ধন হতে পারে।

রিকুয়ারমেন্ট খুঁজে পাওয়ার পরে এই স্পেসিফিকেশনটা চলে যাবে সিস্টেম ডিজাইনার এবং সফটওয়্যার আর্কিটেক্ট এর কাছে। সিস্টেম ডিজাইনার এবং আর্কিটেক্ট কিন্তু দুইটা ভিন্ন পজিশন। সিস্টেম ডিজাইনার এর কাজ হচ্ছে অ্যাপলিকেশনটা যেই সিস্টেমে চলবে তার আর্কিটেকচার তৈরি করা। কত গুলো সার্ভার লাগবে, কোথায় ক্যাশিং লাগবে, ডাটাবেসের ডিজাইনটা কেমন হবে এই রকম গুরুত্বপূর্ণ বিষয় গুলোর ডিসিশন নিয়ে থাকে সিস্টেম ডিজাইনার। এছাড়াও অ্যাপলিকেশনটা কোন আর্কিটেকচার মডেল মেনে তৈরি করা হবে সেই সিদ্ধান্তও নিয়ে থাকে সিস্টেম ডিজাইনার। সফটওয়্যার আর্কিটেক্টের কাজ হচ্ছে কোডের ডিজাইন এবং মডেল তৈরি করা। এমন ভাবে কোড গুলো তৈরি করা যেন কোডগুলো ফিউচার প্রফ হয়। কালকে যদি নতুন কোনো রিকুয়ারমেন্ট আসে, সেই রিকুয়ারমেন্টটা এক্সিস্টিং কোডবেসের সাথে কিভাবে যুক্ত করবে, কিভাবে কোডের ডিজাইন করলে সব থেকে বেশি মডুলার হবে, ফিউচার কোনো আপডেট এক্সিস্টিং কোডকে ব্রেক করতে পারবে না, যদি কোন সমস্যা তৈরিও হয় তাহলে সাথে সাথেই আগের জায়গাতে বোলব্যাক করা যাবে ইত্যাদি ধরনের কাজ গুলো সাধারণত সফটওয়্যার আর্কিটেক্ট করে থাকে।

একটা বিল্ডিং তৈরির পূর্বে যেমন এর আর্কিটেকচার দরকার, ঠিক একটি সফটওয়্যার তৈরির পূর্বও এর আর্কিটেকচার দরকার। একটা বিল্ডিং যেমন কতটা টেকসই হবে সেটা নির্ভর করে কতটা ভালোভাবে আর্কিটেক্ট করা হয়েছে তার ওপরে। একই ভাবে একটা সফটওয়্যারও কতটা নির্ভর যোগ্য, কতটা সিকিউরড, কত মানুষকে একসাথে সার্ভিস দিতে পারবে, এগুলো নির্ভর করে এর সিস্টেম ডিজাইন এবং আর্কিটেকচারের ওপরে। আর্কিটেকচার তৈরি হয়ে গেলেই আমরা

ডেভেলপমেন্ট শুরু করতে পারি। তবে সবার প্রথমে দরকার, আমরা কি ডেভেলপ করতে চাই তার প্রোটোটাইপ। যে কোন প্রোডাক্ট তৈরিব পূর্বে ঠিক ওই রকম দেখতে একটা ডামি প্রোডাক্ট তৈরি করে নেওয়া হয়। সফটওয়্যারের ক্ষেত্রেও এর ব্যতিক্রম নয়।

## System Design and Software Architecture

**System Design:** Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. ([Wikipedia](#))

**Software Architecture:** Software architecture refers to the fundamental structures of a software system and the discipline of creating such structures and systems. Each structure comprises software elements, relations among them, and properties of both elements and relations. The architecture of a software system is a metaphor, analogous to the architecture of a building. It functions as a blueprint for the system and the developing project, laying out the tasks necessary to be executed by the design teams. ([Wikipedia](#))

প্রোটোটাইপ এর কাজ সম্পূর্ণ হওয়ার পরে এটা ডেভেলপমেন্টের জন্য প্রস্তুত হয়ে যায়। একদল মানুষ এর ফ্রন্টেন্ড মানে যা ইউজার দেখে সেই অংশ নিয়ে কাজ করে আর একদল মানুষ এর ব্যাকেন্ড মানে অ্যাপলিকেশনটার মেইন লজিক এবং ডাটা নিয়ে কাজ করে। যখন অ্যাপলিকেশনটা তৈরি হয়ে যায় তখন একদল মানুষ এর কোয়ালিটি অ্যাসিউর করে, টেস্টিং করে। তারপরে আর একদল মানুষ এটাকে প্রোডাকশন সার্ভারে ডেপলয় করে ইউজারদের ব্যবহারের জন্য উন্মুক্ত করে দেয়।

আমি এখানে খুব ছোট করে, সহজ করে বিষয় গুলো বুঝানোর চেষ্টা করলাম। এর মাঝে আরও অসংখ্য কাজ থাকে। তবে বিগিনার হিসেবে আমাদের এত কিছু

---

প্রথমেই জানার দরকার হবে না। একটা অভাবল আইডিয়া থাকলেই আমরা কাজ চালিয়ে নিতে পারব। এখানে আমরা অনেক গুলো ডিফারেন্ট রোলের মানুষের সম্পর্কে জেনেছি। একটা অ্যাপলিকেশন বানাতে এত এত মানুষের দরকার হয়? হ্যাঁ, অনেক মানুষের দরকার হয়। এবং যত মানুষের দরকার হয় ঠিক ততটাই আমাদের অপর্চুনিটি। একজন রিকুয়ারমেন্ট এনালিস্ট থেকে শুরু করে সিস্টেম আর্কিটেক্ট পর্যন্ত অসংখ্য পজিশন আমাদের জন্য অপেক্ষা করছে। ডেভেলপমেন্টই একমাত্র কাজ না। তবে এখানে আমি এত গভীরে যেতে চাচ্ছি না, এখানে শুধু ডেভেলপমেন্ট নিয়েই কথা বলা যাক।

বিভিন্ন ডেভেলপমেন্ট বিভিন্ন রকম, তবে ওয়েব ডেভেলপমেন্টের কথা আসলেই দুইটা টার্ম চলে আসে, আর সেটা হল ফ্রন্টেন্ড এবং ব্যাকেন্ড। ফ্রন্টেন্ডের কাজ হল, একটা অ্যাপলিকেশনের ফ্রন্ট পার্ট মানে ইউজার যা দেখবে সেই অংশটাকে ডেভেলপ করা। আর ব্যাকেন্ড কাজ হল অ্যাপলিকেশনটা কিভাবে চলবে সেই বিষয়টা ডেভেলপ করা। আর যেই ব্যক্তি একই সাথে দুইটা অংশের কাজই করতে পারে তাকে বলা হয় ফুলস্ট্যাক ডেভেলপার। সবাইকে ফুলস্ট্যাক ডেভেলপার হতে হবে এমনটা না। তবে আমি মনে করি একজন ফ্রন্টেন্ড ডেভেলপারের উচিং ব্যাকেন্ড ডেভেলপমেন্ট কিভাবে হয় সেটার একটা নৃন্যতম ধারণা রাখা। আবার একজন ব্যাকেন্ড ডেভেলপারেরও উচিং ফ্রন্টেন্ড সম্পর্কে কিছুটা হলেও জ্ঞান রাখা। তাহলে ডেভেলপমেন্ট প্রসেসটা অনেক সহজ হয়। আপনি যেই কোডটা লিখছেন সেটা অন্য ডেভেলপার কিভাবে ব্যবহার করবে তা জানা থাকলে কোড করতেও অনেক সুবিধা হয়। এছাড়াও বর্তমান বাজারে ফুলস্ট্যাক ডেভেলপারের ডিমান্ড অনেক বেড়ে যাচ্ছে। কোম্পানিগুলো কাজের জন্য আজকাল ফুলস্ট্যাক ডেভেলপারদেরই বেশি প্রাধান্য দিচ্ছে।

ডেভেলপমেন্টে, আপনি যেই অংশেরই কাজ করেন না কেন আপনাকে অনেক বেশি টুলস নিয়ে কাজ করতে হবে। নতুন নতুন টুলস, লাইব্রেরী, ফ্রেমওয়ার্ক প্রতিনিয়তই শিখতে হবে এবং ব্যবহার করতে হবে। প্রথম দিকে যা খুব

---

ভয়ানক লেগে থাকে। যখন আপনার সামনে কয়েকশ টুলস এবং লাইব্রেরী এর কথা কেউ বলবে তখন ভয় লাগাটাই স্বাভাবিক। তবে একটা কথা মাথায় রাখবেন, এই টুলস গুলো কিন্তু এসেছে আপনার জীবনটা সহজ করার জন্য। আপনাকে প্যারা দেওয়ার জন্য নয়। যখন এই টুলস গুলো আপনি শিখে ফেলবেন তখন আগের থেকে অনেক বেশি কাজ আরও সহজে এবং দ্রুত করতে পারবেন। তাহলে শিখতে সমস্যা কোথায়? আবার সব কিছু আপনাকে একবারে শিখতেও হবে না, ব্যবহারও করতে হবে না। আস্তে আস্তে আপনি অনেক কিছু শিখে যাবেন, শুধু শেখার মানুষিকতাটা রাখতে হবে। ডেভেলপমেন্টের এই জগতটাই এমন, প্রতিনিয়ত শেখার ভিতর দিয়েই যেতে হয়। আর এই শেখার প্রোসেসটা মাটেও চালেঙ্গিং না, বরঞ্চ অনেক বেশি এগিয়েবল।

যে কোনো ডেভেলপমেন্টেই আপনি হয়ত কয়েকশ টুলসের একটা লিস্ট পাবেন, যেগুলোর নামও খুব ভয়ানক। যদি আপনার প্রোগ্রামিং স্কিল ভালো থাকে তাহলে এই লিস্টের বেশির ভাগ টুলসই আপনি একদিনে শিখে ফেলতে পারবেন। যেমন কিছু উদাহরণ দেওয়া যাক। JSON, নাম শুনে অনেক ভাবি ভাবি মনে হলও এটা শুধুমাত্র একটা স্পেসিফিকেশন। কিভাবে কিছু ডাটাকে ফর্মেট করে একটা ফাইলে রাখবেন তারই কিছু নিয়ম কানুন দেওয়া আছে এখানে। মাত্র ৭-৮ টা রুলস, যা মনে রাখতে ৩০ মিনিট সময়ই যথেষ্ট। আপনি ব্যয় করলেন মাত্র ৩০ মিনিট সময়, কিন্তু আপনার ঝুলিতে একটা টুলস যুক্ত হয়ে গেল।

এরপরে ধরা যাক Bootstrap এর কথা। এটা একটা CSS ফ্রেমওয়ার্ক। আমরা নিজেরা যখন HTML কোড লিখে CSS ব্যবহার করে স্টাইল করি, তখন ক্লাস সিলেক্টর ব্যবহার করি। মানে HTML এলিমেন্ট ক্লাস নেম বসায়ে রেখে CSS থেকে ক্লাস নেম ধরে সিলেক্ট করি। বুটস্ট্রাপ আমাদের জন্য আগে থেকেই অনেক CSS কোড লিখে রেখেছে এবং আমাদেরকে বিবাট একটা ক্লাস নেমের লিস্ট ধরিয়ে দিয়েছে। আমরা যখন HTML কোড লিখবো তখন এই ক্লাস নেম গুলো ব্যবহার করলে অটোমেটিক স্টাইল হয়ে যাবে। মজার বিষয় হচ্ছে আমাদের ক্লাস

---

নেম গুলোও মনে রাখার দরকার নেই, কারণ Bootstrap এর অফিসিয়াল সাইটে সব কিছুই লিপিবদ্ধ করা আছে। আমাদের শুধু জানতে হবে যে Bootstrap কি কি প্রোভাইড করছে। প্রোগ্রামিং এ দক্ষ যে কোনো ব্যক্তির ১ দিনের বেশি সময় লাগবে না Bootstrap শিখতে। আরও মজার বিষয় হচ্ছে CSS এর সমস্ত ফ্রেমওয়ার্কই এই একই ভাবে কাজ করে। তাই আর তিন চার দিন সময় নিয়ে আরও তিন চারটা জনপ্রিয় ফ্রেমওয়ার্ক দেখে নিলেই আপনার ঝুলিতে আরও কয়েকটা টুলস যুক্ত হতে থাকবে এবং আপনি প্রো হতে থাকবেন।

ডেভেলপমেন্টে আপনাকে প্রতি দিন নতুন নতুন টুলস শিখতে হবে আর এখন আপনি জানেন নতুন টুলস শেখা আসলে বড় কোনো বিষয় না। কিছু টুলস আছে যেগুলো ওয়েব ডিজাইনের সাথে যুক্ত, কিছু টুলস ফ্রন্টেন্ড ডেভেলপমেন্টে কাজে লাগবে, কিছু টুলস আবার ব্যাকেন্ড বা ডেভঅপস এ কাজে লাগবে। কিন্তু কিছু টুলস আছে যে গুলো আপনার সারা জীবনই কাজে লাগবে। এই টুলস গুলো নির্দিষ্ট কোনো ডেভেলপমেন্টের সাথে জড়িত না, বরঞ্চ সব জায়গাতেই ব্যবহৃত হয়। এই কমন টুলস গুলো ডেভেলপমেন্টে যাত্রার শুরুতেই জেনে নেওয়া উচিত, তাহলে যাত্রাটা অনেক সহজ হবে। আমি এখানে যেই টুলস গুলো সম্পর্কে না জানলেই না, শুধুমাত্র সেগুলোই উল্লেখ করলাম।

**কোড এডিটরঃ** একজন প্রোগ্রামার হিসেবে, একজন ডেভেলপার হিসেবে আপনাকে সারা জীবন কোড লিখতে হবে। তাই কোড লেখার জন্য ভালো একটা টেক্সট এডিটর বেছে নেওয়া এবং তার সমস্ত ট্রিক্স, সমস্ত শর্টকাট শিখে নেওয়া খুব জরুরি। এগুলো শিখতে মোটেও সময় লাগবে না। খুব দ্রুত শিখে যাবেন আবার খুব দ্রুত ভুলেও যাবেন। বার বার প্রাকটিস করার মাধ্যমেই এই শর্টকাট ট্রিক্স গুলো মনে রাখতে হয়। গুগলে সার্চ করলে আপনি উজনখানেক ফ্রি কোড এডিটর পেয়ে যাবেন। লোকাল পিসিতে কাজ করার জন্য আমার পছন্দের কোড এডিটর হচ্ছে VSCode যা বর্তমানে সমস্ত ডেভেলপারেরই পছন্দের কোড এডিটর। যারা সার্ভার নিয়ে কাজ করেন তাদের জন্য দরকার হয় টার্মিনাল বেসড

---

টেক্সট এডিটর। এই ক্ষেত্রে আমার পছন্দের তালিকায় প্রথমেই থাকছে VIM এবং দ্বিতীয়তে থাকছে Nano.

**IDE:** যখন লার্জ কোনো প্রজেক্টে কাজ করতে হয় তখন কোড এডিটরের থেকে IDE বা Integrated Development Environment বেছে নেওয়াই ভালো। কারণ এখানে সহজে কোড লেখার সাথে সাথে একটা প্রজেক্ট ম্যানেজ করার সমস্ত ফিচারই দেওয়া থাকে। IDE সম্পর্কে দক্ষ হতে অনেক সময় লাগে কারণ এখানে স্বাভাবিক প্রয়োজনের চেয়েও কয়েক গুণ বেশি ফিচার দেওয়া থাকে। আমার পছন্দের IDE হচ্ছে Jet Brains কোম্পানির সমস্ত IDE, যেমন IntelliJ Idea, Web Storm, PyCharm, CLion. বাজারে অবস্থিত সমস্ত জনপ্রিয় IDE গুলো ব্যবহার করার পরে আমি জেনেছি এদের থেকে বুদ্ধিমান IDE আর কেও তৈরি করে না। তবে দুঃখের বিষয় এদের প্রায় সমস্ত IDE ই পেইড।

**CLI:** একজন সাধারণ কম্পিউটার ব্যবহারকারীর কাছ থেকে যদি কম্পিউটার গ্রাফিক্সকে সরিয়ে নেওয়া হয়, অথবা শুধু মাউসটাই সরিয়ে নেওয়া হয় তার পক্ষে কম্পিউটার চালানো প্রায় অসম্ভব হয়ে দাঁড়াবে। কিন্তু একজন ডেভেলপার শুধু মাত্র টার্মিনাল ব্যবহার করেই কম্পিউটার অপারেট করতে পারে। CLI, মানে Command Line Interface সম্পর্কে একটা বেসিক ধারণা, CLI ব্যবহার করে বেসিক কম্পিউটার অপারেট সম্পর্কে জানা প্রত্যেকটা ডেভেলপারের জন্য একান্ত প্রয়োজনীয় বলে আমি মনে করি। তবে CLI শেখার ক্ষেত্রে linux এর কমান্ড শিখবেন। এর সাথে যদি পারেন Shell বা Bash স্ক্রিপ্টিংটাও শিখে নিবেন যা আপনার কম্পিউটার ইউজেসকে আরও সহজ করবে।

**Git & Github:** ডেভেলপমেন্টের একটা কমন কাজ হচ্ছে কোডের ভাস্রন কন্ট্রোল করা। যখন আপনি একটা কোড রিলিজ দেওয়ার পরে সেই কোডের কোনো আপডেটেড ভাস্রন বের করবেন তখন কাজটা কিছুটা প্যারাদায়ক হয়ে যায়। কারণ নতুন কোডে যদি কোনো ভুল থাকে, তাহলে সাথে সাথেই আগের ভাস্রনে রোল ব্যাক করতে হয়। আর এই প্যারাদায়ক কাজটা সহজ করার জন্য আছে

---

Git, যা এখন ভাস্ন কন্ট্রোলের সাথে সাথে নানা মুখি কাজে ব্যবহৃত হচ্ছে। Git এর একটা ডিস্ট্রিবিউটেড সার্ভার হচ্ছে Github, যা ব্যবহার করা হয় লোকাল কোডকে একটা সেন্ট্রাল সার্ভারে রাখার জন্য। যেন সমস্ত টিম মেম্বাররাই একই কোড শেয়ার করতে পারে। Git হচ্ছে আপনার লোকাল রিপিসিটরি আর Github হচ্ছে রিমোট রিপিসিটরি। এছাড়া Github এরও নানা মুখী ব্যবহার রয়েছে। একজন ডেভেলপার হিসেবে আপনাকে Git এবং এর সঠিক ব্যবহার শিখতেই হবে।

**Chrome Developer Tools:** একজন ওয়েব ডেভেলপার হিসেবে কাজ করতে হলে আপনাকে সব সময়ই কোনো না কোনো ব্রাউজার ব্যবহার করতেই হবে। সাধারণতই ডেভেলপারদের প্রথম পছন্দ হচ্ছে গুগল ক্রোম ব্রাউজারটি। এর পিছনের বড় কারণ হচ্ছে এর ডেভেলপার টুলসের সাপোর্ট। একজন ওয়েব ডেভেলপার হিসেবে গুগল ক্রোমের ডেভেলপার টুলস গুলো সম্পর্কে জানা একান্তই জরুরি। কারণ আপনার অ্যাপ্লিকেশনটি কোন স্টেটে আছে, নেটওয়ার্ক রিকুয়েষ্ট ঠিক ভাবে হচ্ছে কিনা, অ্যাপ্লিকেশনের পার্ফরমেন্স ঠিক আছে কিনা এই সব কিছু আপনি জানতে পারবেন ক্রোমের ডেভেলপার টুলস ব্যবহার করে।

**Agile & Scrum:** ডেভেলপমেন্ট একটা লম্বা প্রসেস। একটা অ্যাপ্লিকেশনের শুরু থেকে শেষ পর্যন্ত অনেক অনেক কাজ থাকে যেগুলো সঠিক ভাবে ম্যানেজ না করলে অ্যাপ্লিকেশনটি সঠিকভাবে ডেভেলপ করাই সম্ভব না। এই জন্য দরকার হয় বিভিন্ন প্রোজেক্ট ম্যানেজমেন্ট টুলস এর। বিভিন্ন সময় বিভিন্ন থিওরি আবিষ্কার হয়েছে, ডেভেলপমেন্ট মডেল তৈরি হয়েছে প্রোজেক্ট গুলো সঠিক ভাবে ম্যানেজ করার জন্য। অসংখ্য মডেলের ভিতরে সব থেকে সফল এবং জনপ্রিয় একটা মডেল হচ্ছে Agile। আর এই মডেলের একটা ফ্রেমওয়ার্ক হচ্ছে Scrum. Scrum ব্যবহার করে একটা প্রোজেক্ট ম্যানেজ করতে শেখাটা খুবই জরুরি একটা বিষয়। এটা শিখতে খুব একটা সময় লাগবে না, কিন্তু সঠিক ভাবে অ্যাপ্লাই করা শিখতে অনেক সময়ের ব্যাপার। তাই আমি মনে করি প্রথম থেকেই

---

অল্ল অল্ল শিখে অল্ল করে প্রাকটিস করা যেতে পারে। তাহলে একটা জনপ্রিয় প্রোজেক্ট ম্যানেজমেন্ট ফ্রেমওয়ার্ক ও শেখা হল, সাথে নিজের প্রোজেক্টও ম্যানেজ করা হল।

ডেভেলপমেন্ট একটা মজার জার্নি, এই জার্নিতে সব কিছুই আছে। এই জার্নিতে ভয় আছে, থ্রিল আছে, মজা আছে আর সব থেকে বেশি আছে নেশা। ডেভেলপমেন্ট, ডেভেলপমেন্টের সাথে জড়িত থাকা ভয়ংকর কিছু নাম, আর রং বেরং এর নতুন নতুন টেকনোলজির প্রতি কেন জানি অজানা একটা নেশা কাজ করে। আপনি না চাইলেও আস্তে আস্তে নেশাগ্রস্থ হয়ে যাবেন আর একবার নেশা গ্রস্থ হয়ে গেলে সফল হওয়ার পরেও থামতে পারবেন না। মনে হবে শুধু শিখেই যাই আর শিখেই যাই। আমাকে অনেকেই প্রশ্ন করে যে, আমি কিভাবে এত দ্রুত নতুন টেকনোলজি শিখতে পারি? আমি পারি কারণ আমি নেশাগ্রস্থ এবং ভিত্তি তৈরি করার সময়ে আমি নিজেকে কোনো ফাঁকি দেইনি। নিজের সাথে কোনো রকমের ছলচাতুরি করিনি। নিজেকে কখনো ধোঁকা দেই নি। আপনি যদি সৎ থাকেন, নিজেকে ধোঁকা না দিয়ে নিজের ভিত্তিটা মজবুত করে গড়ে তোলেন তাহলে আপনিও পারবেন। নিজের ভিত্তি যদি ঠিক থাকে, ডেভেলপমেন্টকে ছেলের হাতের মোয়াই মনে হবে।



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>



---

অধ্যায় পাঁচ

## শিল্প না সাহিত্য?

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ UI/UX ডিজাইন
- ✓ ওয়েব ডিজাইন
- ✓ ফ্রন্টেন্ড ডেভেলপমেন্ট

---

একটা অ্যাপ্লিকেশন মানুষ ব্যবহার করবে কি না সেটা নির্ভর করে অ্যাপ্লিকেশনটা দেখতে কেমন, ব্যবহার করা কর্তৃ সহজ, ইউজারের আকশনের বিপরীতে কর্তৃ ভালো ভাবে ফিডব্যাক দেয় এবং কর্তৃ ভালো ভাবে পার্ফরম করছে, এই সমস্ত বিষয়গুলোর ওপরে। সহজ কথায় বলতে চাইলে অ্যাপ্লিকেশনের ফ্রন্টেন্ডটা কেমন হবে তার ওপরেই নির্ভর করে ইউজার এই অ্যাপ্লিকেশনটা ব্যবহার করবে কি না। সিকিউরিটি বা অন্যান্য বিষয় গুলো সাধারণত পরেই মানুষ চিন্তা করে। তার মনে বলা যেতেই পারে ফ্রন্টেন্ড ডেভেলপাররা একটা বিরাট বড় রোল প্লে করছে এই ক্ষেত্রে। প্রতিটা ফিচার দেখতে কেমন হবে, ইউজার কিভাবে ব্যবহার করবে, কিভাবে ডিজাইন করলে ইউজার সব থেকে সহজে অ্যাপ্লিকেশনটা ব্যবহার করতে পারবে, কি কি ইন্টারেকশন যুক্ত করতে হবে, অ্যাপ্লিকেশনের কালার কেমন হবে, কোন ফন্ট ব্যবহার করা হবে, একটা এলিমেন্ট থেকে আর একটা এলিমেন্টের দূরত্ব কতটুকু হবে এই রকম অসংখ্য গুরুত্বপূর্ণ বিষয়ে সিদ্ধান্ত নিতে হয় একজন ফ্রন্টেন্ড ডেভেলপারকে। কারণ এই সমস্ত বিষয়ের ওপরেই একটা ডিজাইনের সৌন্দর্য নির্ভর করে। একজন ফ্রন্টেন্ড ডেভেলপারকে একই সাথে ডিজাইন সেন্স এবং ডেভেলপমেন্ট সেন্স রাখতে হয়।

ফ্রন্টেন্ড ডেভেলপমেন্টকে আমরা দুই ভাগে ভাগ করে নিতে পারি। প্রথম ভাগে আমরা শুধু ডিজাইন এবং প্রোটোটাইপিং নিয়েই কথা বলব। আর দ্বিতীয় ভাগে ওই প্রোটোটাইপকে কোডে রূপান্তর করার মাধ্যমে কিভাবে ডেভেলপমেন্ট সম্পন্ন করতে হয় সেই বিষয়ে আলোচনা করবো। সাধারণত প্রথম অংশের কাজকে ওয়েব ডিজাইন এবং পরের অংশের কাজকে ফ্রন্টেন্ড ডেভেলপমেন্ট বলা হয়ে থাকে। তবে এই ক্ষেত্রে বিভিন্ন মতবিরোধ রয়েছে। আমরা মতবিরোধের দিকে না গিয়ে প্রথমেই জেনে নিব কিভাবে ডিজাইন প্রসেসটা কাজ করে থাকে, এবং আমরা যদি একজন ওয়েব ডিজাইনার হতে চাই তাহলে আমাদের কি কি শিখতে হবে। এর পরে আমরা ডেভেলপমেন্ট বিষয়ক জ্ঞান অর্জন করবো।

ওয়েব ডিজাইনের কাজটাকে আবার দুইভাগে ভাগ করা যায়। একটা হচ্ছে গ্রাফিক ডিজাইন আর দ্বিতীয়টা হচ্ছে ওয়েব ডিজাইন। একটা সময়, প্রথমে একজন গ্রাফিক ডিজাইনার ওয়েবসাইটটা দেখতে কেমন হবে তা ফটোশপে ডিজাইন করতো। আর একজন ওয়েব ডিজাইনার সেই ডিজাইনটা দেখে দেখে ভুবছ HTML & CSS কোড লিখত। যদিও এটা অনেক পুরাতন পদ্ধতি তারপরেও আমাদের দেশে এই কাজটা অনেক বেশি জনপ্রিয়। এখনো বহু মানুষ PSD to HTML এর এই কাজ করে থাকে। এই পদ্ধতিটা পুরাতন হলেও ছোটেখাটো ওয়েবসাইট তৈরির জন্য খুবই ভাল একটা সমাধান। তবে আপনি যদি ওয়েব অ্যাপ্লিকেশন তৈরি করতে চান তাহলে আমার মনে হয় না, যে এই পদ্ধতিটা খুব একটা ভাল

---

ফলাফল বয়ে আনতে পারে। আর আমি যেহেতু প্রথম থেকে ওয়েব অ্যাপ্লিকেশন নিয়েই কথা বলছি, তাই নতুন যেই সিস্টেমে এই কাজটা করা হয় সেই বিষয়েই আলোচনা করবো।

পূর্বের গ্রাফিক ডিজাইনারের কাজটার এখন নতুন একটা নামকরণ হয়েছে, UI/UX ডিজাইন। এখানে দুইটা ভিন্ন ভিন্ন টার্ম রয়েছে। প্রথমটা হচ্ছে UI - User Interface ডিজাইনার, এনার কাজ হল আপনার ওয়েব অ্যাপ্লিকেশনের ভবগ্রহ একটা কপি তৈরি করা, কিন্তু এখানে কোন ফাংশনালিটি থাকবে না। আপনি পরবর্তীতে এই ডামি অ্যাপ্লিকেশনটা দেখে দেখে আপনার অ্যাপ্লিকেশন তৈরি করবেন। এই কাজটা আপনি যে কোন ডিজাইন বেসড সফটওয়্যার ব্যবহার করেই করতে পারেন, এমনকি ফটোশপ ব্যবহার করেও করতে পারেন। তবে কিছু সফটওয়্যার আছে যেগুলো এই উদ্দেশ্যেই তৈরি করা হয়েছে। এরকম জনপ্রিয় কিছু সফটওয়্যার হচ্ছে Adobe XD, Figma and Sketch. এই সফটওয়্যার গুলো গ্রাফিক সফটওয়্যারের মত এত বিশাল না, অল্প কিছু প্রয়োজনীয় টুলস এখানে আছে, যা ব্যবহার করে আপনি খুব সহজেই অ্যাপ্লিকেশনের জন্য ডামি ইন্টারফেস তৈরি করতে পারবেন। এর সাথে টিম কলাবোরেশনের সুবিধাও পাবেন। এবং এই সফটওয়্যার গুলো শিখতে আপনার গ্রাফিক এর কোন কোর্স করারও দরকার হবে না, নিজে নিজে ঘাটাঘাটি করলে এবং ইউটিউবে কিছু টিউটোরিয়াল দেখলে দুই তিন দিনের ভিতরেই শিখে ফেলতে পারবেন।

এর পরে আসে UX - User Experience ডিজাইনার। আমরা যে ডিজাইন করছি সেটা কতটা লজিক্যাল, কতটা ইউজার স্যাটিস্ফেকশন এনে দেবে, কিভাবে কোন কম্পোনেন্টটা কোথায় রাখলে ইউজার সহজে খুঁজে পাবে, অ্যাপ্লিকেশনের কালার, ফন্ট, ফন্ট সাইজ, অ্যালাইনমেন্ট কেমন হলে ইউজারের অ্যাপ্লিকেশনটা ব্যবহার করতে বিরক্ত লাগবে না, এই রকম সব ক্রিটিক্যাল বিষয়ে রিসার্চ করার কাজ করে থাকে UX Designers. আপনার ব্যবসা সম্পর্কিত যাবতীয় তথ্য জেনে তারপরেই এরা একটা ডিজাইন ডিসিশন তৈরি করে। যেমন যদি আপনার অ্যাপ্লিকেশনটা মেয়েদের জন্য তৈরি হয়, তাহলে এর ডিজাইন একরকম ভাবে হবে। আবার যদি ছেলেদেরকে টাগেটি করে করা হয় তাহলে আর এক রকম হবে। আবার যদি দুইজনকেই টাগেটি করা হয় তাহলে ভিন্ন রকম হবে। আপনার ব্যবসার প্রতিটা বিষয় মাথায় রেখেই UX ডিজাইনাররা একটা অ্যাপ্লিকেশন ডিজাইন করে থাকে। সাধারণত UX ডিজাইনাররাই আপনার জন্য UI ডিজাইন করে থাকে এবং একই সাথে একটা প্রোটোটাইপ ও তৈরি করে ফেলে।

---

একটা অ্যাপ্লিকেশন সফলতার মুখ দেখবে কিনা তার অনেকাংশ নির্ভর করে UI/UX ডিজাইনারের ওপরে। যে কোন একটা অ্যাপ্লিকেশন ডেভেলপমেন্টের পূর্বেই তার UI/UX তৈরি করে নিতে হয়, তা না হলে ওই অ্যাপ্লিকেশনটা ডেভেলপ করা প্রায় অসম্ভব হয়ে দাঁড়ায়। সারা পৃথিবীতেই UI/UX ডিজাইনের মূল্য অনেক বেশি। এমনকি আমাদের দেশেও এর মূল্য দিন দিন বৃদ্ধি পাচ্ছে। আপনি যদি একজন ফুলস্ট্যাক ডেভেলপার হতে চান তাহলে আপনাকে ভাল UI/UX ডিজাইনার হতে হবে না, তবে এর প্রোসেস সম্পর্কে ধারণা থাকলে তা আপনার অনেক কাজে লাগবে। আর আপনি যদি মনে করেন আপনি অ্যাপ্লিকেশন সম্পর্কে ভাল জ্ঞান রাখেন এবং আপনার ডিজাইন সঙ্গে খুবই ভালো তাহলে শুধু মাত্র UI/UX ও আপনার পেশা হতে পারে। বাইরের দেশ গুলোতে UI/UX ডিজাইনারের ইনকাম একজন ফ্রন্টেন্ড ডেভেলপারের মতই এবং কিছু ক্ষেত্রে তার থেকেও বেশি। তবে এই পেশাতে আপনাকে প্রতিনিয়ত প্রচুর ডিজাইন রিলেটেড চ্যালেঞ্জ নিতে হবে।

একজন UI/UX ডিজাইনার হিসেবে ওয়েব সাইটের ইউজার ইন্টারফেস ডিজাইনের সাথে সাথে আপনাকে ইন্টারেকশন ডিজাইনও করতে হবে। ইউজার কোনো একশন ঘটালে কিভাবে অ্যাপ্লিকেশনটি রেসপন্স করবে, দুইটা পেজের ভিতরে ট্রান্সিশন কিভাবে হবে, একটা ইরোর মেসেজ কিভাবে ডিসপ্লে করবে, এই রকম যত ভাবে একটা অ্যাপ্লিকেশনের পক্ষে ইন্টারেক্ট করা সম্ভব সব কিছুই আপনাকে ডিজাইন করতে হবে। কোনো ভাবেই আপনার ইউজার যেন অ্যাপ্লিকেশনটা ব্যবহার করতে বিরক্ত বোধ না করে সেই বিষয়টা মাথায় রাখতে হবে। সহজ ভাবে বললে আপনাকে পুরো ওয়েব অ্যাপ্লিকেশনটিই ডিজাইন করতে হবে। অ্যাপ্লিকেশনে যত গুলো পেজ থাকবে সমস্ত পেজ ডিজাইন করতে হবে। পেজ গুলোর ভিতরে ইন্টার লিংকিং তৈরি করতে হবে। আপনার ডিজাইন করা অ্যাপ্লিকেশনটা দেখতে একদম অরিজিনাল অ্যাপ্লিকেশনের মতই মনে হবে। কিন্তু এখানে কোনো অ্যাপ্লিকেশন লজিক, বিজনেস লজিক থাকবে না। আপনার ডিজাইন করা এই অ্যাপ্লিকেশনকেই বলা হয় প্রোটোটাইপিং (Prototyping) যা বর্তমানে খুবই জনপ্রিয় এবং প্রতিটা অ্যাপ্লিকেশন তৈরির পূর্বেই করা হয়।

## UI/UX ডিজাইন কি

**UI Design:** User interface (UI) design is the process of making interfaces in software or computerised devices with a focus on looks or style.

**UX Design:** User experience (UX) design is the process design teams use to create products that provide meaningful and relevant experiences to users.

## UI/UX ডিজাইনে যা যা শিখতে হবে

### UX Topics

- Business Research
- User Research
- Competitor Research
- Usability Testing

### UI Topics

- Color Theory
- Balance
- Contrast
- Grid System
- Typography
- Consistency

## জনপ্রিয় কিছু UI/UX টুলস

Sketch

Figma

Adobe  
XD

InVision

ProtoPie

# UI/UX শিখার রেফারেন্স

## Must Read UI/UX Books:

- Don't Make Me Think By Steve Krug
- Design in Everyday Life By Don Norman
- Non Designers Design Book By Robin William
- UX Strategy By Jame Levy
- Design Pattern for Design System (Diana Macdonald)
- Inclusive Design for Digital World (Regine Gilbert)

## Youtube Channels to Learn UI/UX

- [Design Course](#) (English)
- [Maex](#) (English)
- [Jessy Showalter](#) (English)
- [UXPin](#) (English)
- [Career Foundry](#) (English)
- [Robert Smith](#) (English)
- [Stack Learner](#) (Bangla) [Upcoming]

## Websites to Get UI/UX Inspiration

- [Awwwards](#)
- [Pttrns](#)
- [UI Movement](#)
- [Collect UI](#)
- [Dribbble](#)

## Training Program for UI/UX

- [Stack Learner Premium Courses](#) [Upcoming]
- [Stack Learner Offline Bootcamps](#)

---

অ্যাপলিকেশনের প্রোটোটাইপ তৈরি হয়ে যাওয়ার পরের কাজটা মূলত ওয়েব ডিজাইনারের। তার কাজ HTML and CSS ব্যবহার করে ডিজাইনের একটা জীবন দান করা। এই কাজ করার জন্য আপনার ভিন্ন ভিন্ন টুলস এর প্রয়োজন পড়বে। একটা অ্যাপলিকেশন ডেভেলপমেন্ট প্রোসেসের মধ্যে সব থেকে সহজ কাজ হচ্ছে এটাই। তবে সহজ কাজ হলেও অনেক গুলো বিষয় আপনাকে জানতে হবে। অনেক থিওরি এবং প্রাক্তিক্যাল কাজের মাধ্যমেই আপনি এই অভিজ্ঞতা অর্জন করতে পারবেন। যেই যেই টুলস এবং ল্যাংগুয়েজ গুলো আপনাকে শিখতে হবে, ছেট্ট করে তার বর্ণনা আমি এখানে দেওয়ার চেষ্টা করছি -

**HTML:** ওয়েব সাইট ডিজাইনের কথা আসলে সবার প্রথমেই যার নাম আসে সেটি হল HTML। আজ ডেভেলপমেন্টের জগত অনেক পরিবর্তিত হয়ে গিয়েছে। নতুন নতুন অনেক টুলস এসেছে, ল্যাংগুয়েজ এসেছে, ডেভেলপমেন্টের থিওরি অনেক পরিবর্তিত হয়েছে কিন্তু HTML পরিবর্তন হয়নি। এটা ঠিক যে, HTML এর নতুন ভাস্বন এসেছে। নতুন নতুন অনেক ফিচার এসেছে, কিন্তু আগেও ওয়েব ডিজাইনের জন্য HTML ব্যবহার করা হতো, এখনো হচ্ছে, ভবিষ্যতেও হবে। আপনি যদি ওয়েবের জগতে নিজের নাম লেখাতে চান তাহলে আপনাকে HTML শিখতেই হবে। HTML এর পূর্ণ রূপ হচ্ছে - Hyper Text Markup Language. এটা একটা মার্কাপ ভাষা। অনেকেই এটাকে প্রোগ্রামিং ল্যাংগুয়েজ মনে করে থাকেন। এটা একটা ভাষা, কিন্তু কোন প্রোগ্রামিং ভাষা না। এটা ব্যবহার করে আমরা বলে দেই যে আমাদের ওয়েব সাইটে কোথায় কি থাকবে, কোথায় বাটন থাকবে, কোথায় লিংক থাকবে, কোথায় ইমেজ থাকবে। আমরা ব্রাউজার ব্যবহার করে ওয়েব সাইট ব্রাউজ করে থাকি। কিন্তু ব্রাউজার এই HTML ব্যাতিত কোন কিছুই বুঝতে পারে না। আমরা যখন ব্রাউজারে গিয়ে ফেসবুকের URL টা লিখে সার্চ করি, ফেসবুকের সার্ভার তখন আমাদের ব্রাউজারকে একটা HTML ফাইল পাঠিয়ে দেয়। ব্রাউজার তখন ওই HTML ফাইলটা পড়ে সেই অনুযায়ী ওয়েব পেজ বের করে আমাদের সামনে দেখায়। আপনি যে কোন ওয়েব সাইটে ভিসিট করে রাইট বাটন ক্লিক করে view page source বাটনে ক্লিক করলে একটা হিজিবিজি লেখা পেজ ওপেন হবে। এখানে আপনি যা দেখতে পাবেন তার বেশিরভাগই হচ্ছে HTML কোড। আর এই HTML কোডটি তৈরি করেছে একজন ওয়েব ডিজাইনার।

HTML খুব সহজ একটা ভাষা যা শিখতে খুব বেশি পরিশ্রম আপনাকে করতে হবে না। এটি কাজ করে ট্যাগ এর মাধ্যমে। যদি আপনি কোন একটি লেখাকে টাইপেল বানাতে চান তাহলে সেই লেখাটার সামনে <h1> ওপেনিং ট্যাগ এবং শেষে </h1> ক্লোসিং ট্যাগ দিতে হবে। এতেই

---

বাউজার বুঝে যাবে যে কিভাবে লেখাটা দেখাতে হবে। এরকম বিভিন্ন কাজের জন্য বিভিন্ন রকমের ট্যাগ রয়েছে। আপনার কাজ এই ট্যাগ গুলো সম্পর্কে জ্ঞান অর্জন করা, কোন ধরনের কাজের জন্য HTML কোন ধরনের ট্যাগ প্রোভাইড করছে সেটা জেনে নেওয়া। একটা রিয়েল ওয়েবসাইট ডিজাইন করতে এরকম হাজার হাজার ট্যাগ আপনাকে নির্দিষ্ট ভাবে ব্যবহার করতে হবে।

**CSS:** HTML ব্যবহার করা হয় একটা ওয়েবসাইটের কাঠামো তৈরি করার জন্য, কিন্তু এই কাঠামো টা মাটেও সুদর্শন না। একটা ওয়েবসাইটের চেহারা সুন্দর করার জন্য ব্যবহার করা হয় Cascading Style Sheet বা যাকে আমরা CSS বলে থাকি। এটাও একধরনের ভাষা, তবে প্রোগ্রামিং ভাষা নয়। CSS আপনি HTML ফাইলের ভিতরে লিখতে পারেন, এমনকি HTML এর ট্যাগের ভিতরেও লিখতে পারেন। তবে সব থেকে ভাল উপায় হচ্ছে নতুন একটা ফাইল তৈরি করে সেখানে CSS কোড লেখা এবং HTML এর সাথে কাজ করার জন্য ফাইলটাকে HTML ফাইলের সাথে লিংক করে দেওয়া। CSS মূলত কাজ করে সিলেক্টরের মাধ্যমে। HTML ট্যাগ বা এলিমেন্ট গুলোকে সিলেক্ট করার জন্য অনেক ধরনের সিলেক্টর আছে। প্রতিটা এলিমেন্ট আলাদা আলাদা ভাবে সিলেক্ট করে আপনার মন মত সব স্টাইল রুলস লিখে দিতে পারেন। কিভাবে স্টাইল রুলস লিখেবন, কি স্টাইল রুলস লিখেবন তা আগে থেকেই CSS বলে দিয়েছে। আপনার কাজ শুধু সেই রুলস গুলো শিখে নেওয়া।

CSS খুবই সহজ একটা ল্যাংগুয়েজ যা শিখতে আপনার HTML এর থেকেও কম সময় লাগবে। কিন্তু সহজ হলেও এর প্রচুর স্টাইল রুলস আছে। যেগুলো মনে রাখতে কিছুটা সময় আপনার ব্যয় হবে। আপনি শুধুমাত্র CSS ব্যবহার করেই একটা সিম্পল ওয়েবপেজকে সর্বোচ্চ সৌন্দর্যের কাতারে নিয়ে যেতে পারেন। এনিমেশন, রেস্পন্সিভ ডিজাইনের মত কাজ গুলোও আপনি শুধুমাত্র CSS ব্যবহার করেই করতে পারেন। তাই CSS সহজ হলেও এটাতে মাছার হতে কিন্তু অনেক সময় এবং অভিজ্ঞতা লেগে যায়।

**CSS Architecture:** CSS হচ্ছে একটা স্পেসিফিকেশন টাইপের ল্যাংগুয়েজ। এর মাধ্যমে আমরা ধরে ধরে প্রতিটা এলিমেন্টের চেহার কেমন হবে বলে দেই। যার ফলে একটা ছোট ওয়েব সাইটের ডিজাইন করলেও দেখা যায় হাজার হাজার লাইনের CSS কোড লেখা হয়ে গেছে। হাজার লাইনের কোড হওয়াটা কোন সমস্যা না, সমস্যা হচ্ছে এই কোড গুলোকে সঠিক ভাবে ম্যানেজ করতে পারাটা। CSS কোডকে সঠিক ভাবে ম্যানেজ করতে আমাদের কিছু কোড আর্কিটেকচার মনে চলা উচিত। যেই আর্কিটেকচার গুলো অনেক বছর ধরে CSS

---

কোড ম্যানেজ করার জন্য ব্যবহৃত হচ্ছে। এরকম কিছু আর্কিটেকচার হল - OOCSS (Object Oriented CSS), SMACSS (Scalable and Modular Architecture for CSS), ITCSS (Inverted Triangle CSS), ACSS (Atomic CSS), BEM (Block Element Modifier) ইত্যাদি। ওয়েব সাইট ডিজাইন করার জন্য কোনো রকম কোনো আর্কিটেকচারের দরকার নেই, আপনি সঠিক CSS কোড লিখলেই ব্রাউজার বুঝতে পারবে। কিন্তু আপনি যদি বড় কোনো অ্যাপলিকেশন ডিজাইন করে থাকেন, সেখানে সঠিক ভাবে কাজ সম্পন্ন হওয়ার সাথে সাথে আপনার কোডটা ম্যানেজেবল কিনা সেটাও অনেক গুরুত্বপূর্ণ বিষয়। আর যখন আপনি কোনো আর্কিটেকচার মডেল ফলো করে কোড করবেন তখন অনেক কম কোড লিখে অনেক বেশি পরিমাণ কাজ করতে পারবেন। তাই যদিও এই মডেল গুলো শিখতে কিছু সময় এবং শ্রম লাগবে তারপরেও চেষ্টা করবেন এগুলো শিখে নেওয়ার, যা আপনাকে অন্য ডিজাইনারের থেকে অনন্য করে তুলবে।

**CSS Preprocessors:** আমরা জানি CSS কোন প্রোগ্রামিং ল্যাংগুয়েজ না। তাই CSS ব্যবহার করে যখন আমরা কোড করি তখন অনেক বেশি কোড আমাদেরকে লিখতে হয়। একবার কোড লিখলেই যদি কাজ হয়ে যেত তাহলেও আমাদের তেমন কোন সমস্যা ছিল না। কিন্তু চিন্তা করেন, একটা ওয়েব সাইটের জন্য আপনি কয়েক হাজার লাইনের কোড লিখলেন কিন্তু তা আপনার ফ্লাইন্টের পছন্দ হল না। তখন এই হাজার হাজার লাইনের কোড আবার আপডেট করতে হবে যেটা খুবই কষ্টদায়ক কাজ। CSS এর ভিতরে যদি প্রোগ্রামিং ল্যাংগুয়েজের কিছু ফিচার যুক্ত করা যেত তাহলে আমরা আরও সহজে, আরও এফিসিয়েন্ট ভাবে CSS কোড লিখতে পারতাম।

ঠিক এই সমস্যাটার সমাধানই করছে CSS Preprocessors. এটা কিন্তু CSS না, এটা একটা কম্পাইলারের মত কাজ করে। বিভিন্ন রকম প্রিপ্রেসর আছে যাদের প্রত্যেকের CSS লেখার নিজস্ব রুলস এবং সিনট্যাক্স আছে। আপনি সেই রুলস ব্যবহার করে CSS কোড লিখবেন। CSS কোডের সাথে সাথে এখানে আরও এক্সট্রা কিছু ফিচার এবং সিনট্যাক্স আছে যা ব্যবহার করে আপনি প্রোগ্রামিং এর মত করে CSS লিখতে পারবেন। কিন্তু সমস্যা হচ্ছে এই সিনট্যাক্স গুলো আমাদের ব্রাউজার বুঝতে পারে না। আমাদের ব্রাউজার শুধুমাত্র Plain CSS কোড বুঝতে পারে। একটা কম্পাইলার এই এক্সট্রা সিনট্যাক্স এবং ফিচার সহ লেখা কোড গুলোকে সাধারণ CSS কোডে রূপান্তর করে ফেলে। আর তখন আমাদের ব্রাউজার এই সিনট্যাক্স গুলো বুঝতে পারে। ছোটোখাটো প্রজেক্টের জন্য কোন Preprocessors এর ব্যবহার করার দরকার না হলও যখন আপনি অনেক পেজ আছে

---

এমন কোন ওয়েবসাইটের ডিজাইন করবেন তখন এটা আপনাকে অনেক সাহায্য করবে। আপনার কোড লেখার সময় অনেক বেঁচে যাবে এবং পরবর্তীতে যদি কোন আপডেটও করতে হয় তাহলেও কোন রকম কোন প্যারা ছাড়াই আপনি পুরো ওয়েব সাইটের আপডেট করতে পারবেন। এমনটা না যে Preprocessor না জানলে আপনি ওয়েব ডিজাইনার হতে পারবেন না। আমি আগেই বলেছি, ডেভেলপমেন্টে আমরা অসংখ্য টুলস, ল্যাংগুয়েজ, লাইব্রেরী এবং ফ্রেমওয়ার্কের সাথে পরিচিত হব। প্রথম দেখায় যেগুলোকে দেখে অহেতুক মনে হতে পারে, কিন্তু এর সব কিছুই এসেছে আমাদের কোন না কোন সমস্যার সমাধান করার জন্য।

বাজারে অনেক রকমের Preprocessors থাকলেও সব থেকে জনপ্রিয় কয়েকটা Preprocessor হচ্ছে Sass, Less and Stylus। আপনি যে কোনো একটা Preprocessor যদি শিখতে চান তাহলে আমি আপনাকে সাজেস্ট করবো Sass শেখার জন্য, কারণ এর জনপ্রিয়তা সব থেকে বেশি এবং বেশির ভাগ কোম্পানি এটাই ব্যবহার করছে। Sass শিখতে আপনার খুব বেশি সময় লাগবে না। যদি আপনি প্রোগ্রামার হয়ে ডেভেলপমেন্ট করতে আসেন, তাহলে কয়েক ঘণ্টার ভিতরেই আপনার CSS লেখার স্টাইল পাল্ট ফেলতে পারবেন Sass শেখার মাধ্যমে। আর Sass শেখা হয়ে গেলে অন্য যে কোন Preprocessor আপনি তৃতী মেরেই শিখে ফেলতে পারবেন। এমনকি আগে থেকে না শিখে শুধুমাত্র তার ডকুমেন্টেশন দেখেই কাজ করতে পারবেন।

**Responsive Web Design:** বাজারে যখন থেকে মোবাইল ফোনে ইন্টারনেট ব্যবহারকারীর সংখ্যা বৃদ্ধি পেয়েছে তখন থেকেই সবাই রেস্পন্সিভ ডিজাইনের প্রতি ঝুঁকেছে। বর্তমানে বেশির ভাগ সাইট ডিজাইন করা হয় মোবাইল ফার্স্ট থিওরি মেনে। মানে প্রথমে মোবাইল ফোনের জন্য ডিজাইন করে পরবর্তীতে ডেক্সটপ বা অন্যান্য বড় ডিভাইজের জন্য ডিজাইন করা হয়। রেস্পন্সিভ ডিজাইনের অর্থ আপনার ওয়েবসাইট বিভিন্ন সাইজের মনিটরের জন্য ডিন ভাবে রেসপন্স করবে। মনে করেন আপনি একটা ওয়েব সাইট ডিজাইন করলেন যা ১৫ ইঞ্চির ল্যাপটপ স্ক্রিনে খুব ভাল দেখা যাচ্ছে, কিন্তু সেই একই ওয়েবসাইট কি ৫ ইঞ্চির বা ৫০ ইঞ্চির স্ক্রিনে ভাল লাগবে না। তারমানে যদি আপনি ওয়েব সাইটকে রেস্পন্সিভ না করেন তাহলে শুধু মাত্র যাদের কম্পিউটার স্ক্রিন ১৫ ইঞ্চি তারাই সব থেকে ভালোভাবে আপনার ওয়েবসাইটটা ব্যবহার করতে পারবে, বাকি ইউজাররা মোটেও মজা পাবে না। তারমানে রেস্পন্সিভ না করলে আপনি বিরাট একটা অভিয়েন্স হারিয়ে ফেলবেন।

---

রেস্পন্সিভ ডিজাইন করার জন্য আপনাকে নতুন করে কিছু শিখতে হবে না, শুধুমাত্র CSS ব্যবহার করেই আপনি যে কোন ওয়েবসাইটকে রেস্পন্সিভ করতে পারবেন। এর জন্য আপনাকে CSS এর Media Query এবং Responsive Measurement Units সম্পর্কে জানতে হবে। তবে এটা খুবই বিরক্তকর একটা কাজ। কারণ একটা ওয়েবসাইট আপনাকে ভিন্ন ভিন্ন ৫-৭ টাইপের স্ক্রিনের জন্য ভিন্ন ভিন্ন ভাবে ডিজাইন করতে হবে। এবং এটা ভালোভাবে করার জন্য অবশ্যই আপনাকে অনেক দিন কাজ করে অভিজ্ঞতা অর্জন করতে হবে।

**CSS Frameworks:** প্রথম প্রথম CSS কোড লিখতে ভাল লাগলেও যখন আপনি প্রফেশনাল ভাবে কাজ করবেন তখন আর মোটেও হাজার হাজার লাইনের CSS কোড লিখতে ভালো লাগবে না। আর প্রতিটা প্রোজেক্টের জন্যই একই ধরনের CSS প্রতিনিয়ত লেখাটাও সময় নষ্ট করা ছাড়া আর কিছুই না। এই জন্য ভালো হয় একটা নির্ভরযোগ্য CSS ফ্রেমওয়ার্ক শিখে নিলে। প্রথমত প্রচুর CSS কোড লেখার হাত থেকে বাঁচা যাবে আর দ্বিতীয়ত ডিজাইন সম্পর্কে কোন জ্ঞান না থাকার পরেও সুন্দর সুন্দর ওয়েবসাইট ডিজাইন করা যাবে। ফ্রেমওয়ার্ক শেখার আর একটা বড় কারণ হতে পারে রেস্পন্সিভ ডিজাইন। কিছু কিছু ক্ষেত্রে ডিজাইনাররা শুধুমাত্র রেস্পন্সিভ ফিচারটার জন্যই CSS ফ্রেমওয়ার্ক ব্যবহার করে থাকে। ফ্রেমওয়ার্ক ব্যবহার করলে ওয়েবসাইট রেস্পন্সিভ করার জন্য প্রচুর CSS কোড লেখার আর দরকার হবে না।

CSS ফ্রেমওয়ার্কের যখন কথা আসে তখন অনেকেই অনেক ফ্রেমওয়ার্কের সাজেশন দিয়ে থাকেন। কেউ বলেন এটা ভালো তো কেউ বলেন এটা। আসলে সব ফ্রেমওয়ার্কই ভালো, আর সব গুলো প্রায় একই ওয়েতে কাজ করে। তাই একটা ভালোভাবে শিখতে পারলে বাকি গুলোতেও কাজ করতে পারবেন, শুধু চাঁথের সামনে তার ডকুমেন্টেশনটা খোলা থাকলেই হবে। CSS ফ্রেমওয়ার্ক বেছে নেওয়ার সময় কিছু বিষয় মাথায় রাখতে হবে, যেমন - রেস্পন্সিভ গ্রিড সিস্টেম আছে কিনা, বিডেল্টইন কম্পোনেন্ট কত গুলো আছে এবং সে গুলো দেখতে কেমন, ইউটিলিটি সাপোর্ট আছে কিনা, কাস্টমাইজেশন করা যায় কিনা, ফাইলের সাইজ কত, কমিউনিটি সাপোর্ট কেমন? CSS ফ্রেমওয়ার্কের কথা বললে, এবং সব থেকে বেশি সুযোগ সুবিধার কথা বললে, সবার প্রথমে Bootstrap এর নামটাই চলে আসে। এটা খুবই জনপ্রিয় এবং বহুল ব্যবহৃত ফ্রেমওয়ার্ক। তাই চাঁথ বক্স করে আপনি এটা শিখে নিতে পারেন। আর এটা শেখা হয়ে গেলে বাকি আরও যেসব ফ্রেমওয়ার্ক আছে যেমন, Bulma, W3CSS,

---

Semantic UI, এরকম হাজার হাজার নাম না জানা ফ্রেমওয়ার্ক আপনি না শিখেই ব্যবহার করতে পারবেন।

**JQuery:** আপনার ওয়েব সাইটে যদি কিছু ইন্টারেকশন যুক্ত করতে চান তাহলে সব থেকে সহজ সমাধানই হল JQuery. এটা জাভাস্ক্রিপ্টের একটা লাইব্রেরী এবং ব্যবহার করতে খুব বেশি জাভাস্ক্রিপ্ট জানারও প্রয়োজন হয় না। কারণ প্রচুর পরিমাণ JQuery Plugins আছে যা খুব সহজেই ব্যবহার করা যায় এবং স্পেশিয়াল সব ইফেক্ট যুক্ত করা যায়। এই ক্ষেত্রে অনেকেই মনে করেন যে JQuery শেখার কোনো দরকার নেই। আমি মনে করি কোনো কিছু শিখলে কোনো ক্ষতি নেই, বরঞ্চ আপনার স্কিল লিস্টে নতুন কিছু যুক্ত হবে। তবে আপনি যদি ভেবে বসে থাকেন JQuery শিখলেই সব হয়ে যাবে তাহলে আপনি অনেক বড় ভুল করবেন। শুধুমাত্র ওয়েব ডিজাইনের ক্ষেত্রেই এটি অনেক বড় একটা ভূমিকা রাখে। আর JQuery এর সর্বোচ্চ ব্যবহার করার জন্য আপনাকে ভালোভাবে জাভাস্ক্রিপ্টও শিখতে হবে।

আপনি যদি এই পর্যন্ত শিখতে পারেন তাহলেই আপনাকে ওয়েব ডিজাইনার বলা হবে। ওয়েব ডিজাইনারের কাজ যে কোন একটা ওয়েব সাইটের ডিজাইন HTML and CSS ব্যবহার করে তৈরি করা। এবং এই বিষয় গুলো শেখার জন্য আমার সাজেশন হল w3school এবং mozilla developer network এর ওয়েবসাইট। আমার মনে হয়না HTML এবং CSS শেখার জন্য এর থেকে ভাল কোন রিসোর্স হতে পারে বা দরকার আছে। বিগিনারদের ভিতরে একটা প্রবণতা আমি সচরাচর লক্ষ্য করে থাকি, যে তারা ডকুমেন্টেশন পড়তে চায় না। ঘন্টার পর ঘন্টা সময় তারা ইউটিউবে টিউটোরিয়াল দেখে নষ্ট করে তাও ডকুমেন্টেশনের ধারে কাছে যেতে চায় না। আমি বিষয়টা খোলাখোলি ভাবে বলতে পারছি কারণ আমি নিজেও এই দলেরই একজন সদস্য ছিলাম। কিন্তু আমাদের মনে রাখতে হবে ডকুমেন্টেশন হচ্ছে যারা এই টুলসটা তৈরি করেছে তাদের প্রোভাইড করা দলিল, আর তাদের থেকে তো বেশি আর কারোরই জানার কথা না, তাই না? এই জন্য ডেভেলপমেন্টে চিকে থাকতে গেল আমাদের প্রচুর ডকুমেন্টেশন পড়তে হবে। প্রতিদিনই আমাদের নতুন নতুন কোন না কোন টুলস বা লাইব্রেরী নিয়ে কাজ করতে হয়। সব কিছু কি একদিনে মনে রাখা সম্ভব? আর সব কিছু কি মনে রাখার দরকারও আছে? আপনি যদি ডকুমেন্টেশন পড়তে জানেন, তাহলে কোন কিছুই আপনার মাথায় রাখতে হবে না। এবং প্রতিনিয়ত নতুন নতুন টুলস নিয়ে খেলা করবেন, তাতেও বোরিং লাগবে না বরঞ্চ অনেক বেশি মজা পাবেন।

## HTML এ যা শিখতে হবে

- How does HTML Work
- HTML Elements & Attributes
- HTML Head and Meta Tags
- Heading, Paragraph & Formatting
- HTML Links & Images
- HTML Lists & Tables
- HTML Forms
- HTML Graphics
- HTML Media
- HTML5 and It's API

## CSS এ যা শিখতে হবে

- CSS Selectors
- CSS Values & Units
- Text & Font Styles
- CSS Box Model
- Style Link, List & Tables
- CSS Display & Positions
- Pseudo Classes & Elements
- CSS Layout using Float & Div
- Flexbox and Grid System
- Responsive Design & Media Query
- CSS Animations and Effects
- Style Media Elements

## SASS এ যা শিখতে হবে

- Sass Installation
- Sass Variables
- Sass Nesting
- Sass @import
- Sass @mixin
- Sass @extend
- Sass String & Numeric Functions
- Sass List Function
- Sass Map Function
- Sass Selector
- Sass Introspection
- Sass Color

## Bootstrap এ যা যা শিখতে হবে

- Bootstrap Grid System
- Responsive Layout Classes
- Bootstrap Theming
- Bootstrap Utilities
- Images, Tables & Figures
- Basic Components
- Bootstrap Cards and Modal
- Carousel, Collapse & Dropdown
- Bootstrap Forms & Inputs
- Bootstrap List Group
- Bootstrap Nav & Navbar
- Extend Bootstrap

## JQuery এ যা যা শিখতে হবে

- JQuery Selectors
- JQuery DOM Traversing
- JQuery DOM Manipulation
- JQuery Events
- JQuery Forms
- JQuery Effects
- JQuery Collections
- JQuery AJAX
- JQuery Utilities
- JQuery Plugins

## কিছু গুরুত্বপূর্ণ ক্রোম এন্টেনশন এবং টুলস

- [Dom Flags](#)
- [Sizzy](#)
- [CheckBot](#)
- [GistBox Clipper](#)
- [Wappalyzer](#)
- [VisBug](#)
- [Web Developer](#)
- [What Font](#)
- [Color Zilla](#)
- [Awesome Screenshot](#)

# HTML, CSS and Bootstrap শিখার রেফারেন্স

## Must Read HTML, CSS & Bootstrap Books:

- HTML & CSS: Design and Build Web Sites By John Duckett
- Web Design with HTML, CSS, Javascript and JQuery Set By John Duckett
- Responsive Web Design with HTML5 and CSS3 By Ben Frain
- Beginning HTML5 and CSS3 By Appress Publication
- Mastering Bootstrap 4 By Benjamin Jakobus
- Bootstrap 4 Quick Start: A Beginner's Guide to Building Responsive Layouts with Bootstrap 4 By Jacob Lett

## Youtube Channels to Learn HTML & CSS

- [Moshiur](#) (Bangla)
- [Anisul Islam](#) (Bangla)
- [Stack Learner](#) (Bangla)
- [Design Course](#) (English)
- [Traversy Media](#) (English)
- [Dev Tips](#) (English)

## Websites to Learn HTML & CSS

- [W3Schools](#)
- [Mozilla Developer Networks](#)
- [Get Bootstrap](#)
- [JQuery Official Documentation](#)
- [CSS Tricks](#)

## Training Program for HTML & CSS

- [Stack Learner Premium Courses](#)
- [Stack Learner Offline Bootcamps](#)

---

এতক্ষণ পর্যন্ত আমরা যা কিছু নিয়ে আলোচনা করলাম তার সবটাই ওয়েব ডিজাইন। এখানে ভিন্ন ভিন্ন দুইটা ক্যারিয়ার আপনি গড়তে পারেন, যার প্রথমটা হল UI/UX Designer এবং দ্বিতীয়টা হল Web Designer, তবে আপনি চাইলে দুইটা একসাথেও করতে পারেন এবং এই ক্ষেত্রে আপনার ভ্যালু এবং ডিমান্ড দুটোই বৃদ্ধি পাবে। এবার আসা যাক ফ্রন্টেন্ড ডেভেলপমেন্ট। এই ক্ষেত্রেও আপনাকে একটা অ্যাপ্লিকেশনের ফন্ট বা ইউজার যেই অংশটা ব্যবহার করবে সেই অংশটাই তৈরি করতে হবে। তাহলে একটু আগে যে বললাম এটা ওয়েব ডিজাইন, এখন আবার বলছি এটা ফ্রন্টেন্ড ডেভেলপমেন্ট। আসলে দুইটার মধ্যে পার্থক্যটা কোথায়?

পূর্বে বা এখনো যখন আমরা একটা ওয়েবসাইট ডিজাইন করি, সেই ক্ষেত্রে ফন্টসাইটে তেমন কোন কাজ বা ইন্টারেকশন থাকে না। এই ধরনের সাইট গুলোর মাধ্যমে আমরা শুধুমাত্র কিছু ইনফরমেশন শেয়ার করে থাকি। ইউজার ওয়েবসাইটে ভিসিট করে, আর্টিকেল পড়ে, সর্বোচ্চ হলে একটা লাইক কমেন্ট করে। এর থেকে বেশি কাজ সাধারণত ওয়েবসাইটের ক্ষেত্রে থাকেনা। তাই এই ধরনের ওয়েবসাইট ডিজাইনের জন্য স্পেশিয়াল কোন টুলস বা টেকনিকও দরকার হয় না। সাধারণ HTML, CSS এবং কিছু কিছু ক্ষেত্রে অল্প একটু Javascript ব্যবহার করলেই হয়ে যায়। এইগুলোকে বলা হয় ওয়েবসাইট, আর ওয়েবসাইটের ফ্রন্টেন্ড কোন ডেভেলপমেন্টের দরকার হয় না।

কিন্তু বর্তমানে ওয়েব বেসড অ্যাপ্লিকেশন গুলোর ডিমান্ড বেড়ে গেছে। এখন ইউজাররা ওয়েব ব্যবহার করে শুধু তথ্য দেখতে প্রস্তুত নয়। তারা প্রচুর ইন্টারেকশন চায়, সমস্ত কাজ ওয়েবে বসেই করতে চায়। আমি এই যে এখন বইটা লেখছি, কয়েক বছর আগে হলেও মাইক্রোসফট ওয়ার্ড পিসিতে ইন্সটল করে আমাকে লিখতে হতো। কিন্তু আমি লেখছি ব্রাউজারে গুগল ডকস ব্যবহার করে। যদিও গুগল ডকস একটা ওয়েব সাইট তারপরেও এখানে একটা অ্যাপ্লিকেশনের সমস্ত ফিচার রয়েছে। আমরা সারাদিন ফেসবুকে বসে যে লাইক কমেন্ট এর বন্যা বইয়ে দেই, শেয়ার করে, ভিডিও দেখে ফাটিয়ে দেই সেই ফেসবুক কিন্তু কোন ওয়েবসাইট না, এটা একটা অ্যাপ্লিকেশন। ওয়েবসাইট হল যেখানে আমরা শুধু তথ্য পাব, নিজে থেকে কোন কিছু করার সুযোগ স্থানে নেই। আর ওয়েব অ্যাপ্লিকেশন হল একটা অ্যাপ্লিকেশন, যেখানে তথ্য পাওয়ার সাথে সাথে প্রচুর ফিচার রয়েছে। অনেক অনেক কাজ স্থানে করা যায়, কিন্তু এটা ওয়েবের মাধ্যমে সার্ভ করা হচ্ছে। এখন আপনারা খেয়াল করলেই দেখতে পারবেন আমরা প্রতিদিন কয়টা ওয়েবসাইট ভিসিট করি আর কয়টা ওয়েব অ্যাপ্লিকেশন ব্যবহার করে থাকি?

---

ফেসবুক, টুইটার, লিঙ্কডিন, ইউটিউব এরকম যত সোশ্যাল অ্যাপ আছে সবই ওয়েব অ্যাপলিকেশন। লেখালেখি করার জন্য বা প্রজেক্টেশন তৈরি করার জন্য যে আমরা গুগল ডকস, গুগল শীট, গুগল স্লাইড ব্যবহার করে থাকি এইগুলো হল ওয়েব অ্যাপলিকেশন। আমরা যেই মেইল ব্যবহার করে থাকি, সেটাও একটা ওয়েব অ্যাপলিকেশন। এখন বাজারে ফটো এডিটিং, ভিডিও এডিটিং করার হাজার হাজার ওয়েব অ্যাপলিকেশন রয়েছে। শুধু তাই না, আপনি ব্রাউজারে বসে মাল্টিমিডিয়ার গেমও খেলতে পারবেন।

বর্তমানে সফটওয়্যারের বিজনেস মডেল পরিবর্তিত হয়ে গেছে। পূর্বে একটা সফটওয়্যার সেল হতো এবং একবার সেল করার পরে কোম্পানি গুলোকে কয়েক বছর ফ্রিতে আপডেট দিতে হতো। যার ফলে তাদের সফটওয়্যারের দাম অনেক বেশি রাখতে হতো। আর দাম বেশি হওয়ার কারণে তাদের ইউজারও তুলনামূলক কম হতো। এর সাথে সাথে সফটওয়্যার ক্রাকিং এর সাথে তে আমরা সবাই পরিচিত। দামি দামি সব সফটওয়্যার কিছু কোম্পানি ক্র্যাক করে কম মূল্যে বিক্রি করে নিজেরা তো অনেক অর্থ উপার্জন করতো, কিন্তু যেই কোম্পানি এই সফটওয়্যারটা বানিয়েছে তাদের ভাঁড়ে মা ভবানী। এই সমস্ত কারণে আমরা সফটওয়্যারের নতুন বিজনেস মডেল পেয়েছি। আর তা হচ্ছে সাবস্ক্রিপশন মডেল।

অন্ন কিছু বড় বড় সফটওয়্যার ব্যতীত যেই সফটওয়্যার গুলো ওয়েব প্লাটফর্মে কোন রকম সমস্যা ছাড়াই চালানো সম্ভব সেই সমস্ত সফটওয়্যার এখন আমরা ওয়েব অ্যাপলিকেশন হিসেবে ব্যবহার করি। তারা এখন একবারে সফটওয়্যার সেল না করে মাসিক একটা ফি নিয়ে ভাড়া দেয়। আপনি ভাড়া দিলে সফটওয়্যারটি চালাতে পারবেন, না দিলে পারবেন না। আর যেহেতু এটা একটা ওয়েব অ্যাপলিকেশন তাই, সফটওয়্যারটি অন্য কেউ ক্র্যাক করেও চালাতে পারবে না। প্রতি মাসেই ইউজারের কাছ থেকে সাবস্ক্রিপশন ফি বাবদ একটা টাকা যেহেতু কোম্পানি গুলো পায়, তাই তাদের সফটওয়্যারের দামও আগের মত এত বেশি রাখার দরকার হয় না। আর দাম যেহেতু অনেক কম, মাসে ১-২০ ডলার তাই প্রচুর ইউজার এখন অ্যাপলিকেশন গুলো ব্যবহার করছে। এটাকে বলা হয় win win পলিসি। কোম্পানি দাম কম রেখেও তাদের কোন লস হচ্ছে না, আবার ইউজারও প্রতিমাসে অন্ন অন্ন করে টাকা দিয়ে অ্যাপলিকেশনটা তার সাধ্যের মধ্যেই ব্যবহার করতে পারছে।

ওয়েব অ্যাপলিকেশনের ফ্রন্টেন্ড তৈরি করা, আর ওয়েবসাইটের ফ্রন্টেন্ড তৈরি করা এক বিষয় না। ওয়েব অ্যাপলিকেশনের ফ্রন্টেন্ড তৈরির জন্য অনেক বেশি জ্ঞান এবং দক্ষতার প্রয়োজন হয়। আর যারা ওয়েব অ্যাপলিকেশনের ফ্রন্টেন্ড ডিজাইন করে, ডেভেলপ করে তাদেরকে

---

ফন্টেন্ড ডেভেলপার বলা হয়। ফন্টেন্ড ডেভেলপারের দায়িত্বটা আরও ভালভাবে বুঝতে চাইলে আমাদের মোবাইল ডেভেলপারদের কাজটা বুঝতে হবে। একজন মোবাইল ডেভেলপারকে তা আমরা কখনোই ছোট করে দেখতে পারি না, তাদের আমরা ডেভেলপার বলেই সম্মান করি। একজন মোবাইল ডেভেলপার আর একজন ফন্টেন্ড ডেভেলপারের মধ্যে পার্থক্য হল, মোবাইল ডেভেলপার শুধুমাত্র একটা ছোট্ট স্ক্রিনের জন্য অ্যাপলিকেশন ডেভেলপ করে। আর একজন ফন্টেন্ড ডেভেলপার ওই একই অ্যাপলিকেশন ওয়েবের জন্য মাল্টিপ্ল স্ক্রিনের জন্য ডেভেলপ করে থাকে। যদি সেই অর্থে বলা যায়, তাহলে একজন ফন্টেন্ড ডেভেলপার মোবাইল অ্যাপলিকেশন ডেভেলপারের সমান এবং কিছু কিছু ক্ষেত্রে অনেক বেশি পরিমাণ দায়িত্ব পালন করে থাকে।

একজন ভাল মানের ফন্টেন্ড ডেভেলপার হতে চাইলে আপনাকে অনেক কিছুই শিখতে হবে, অনেক ধৈর্য নিয়ে অনেক পরিশ্রম করতে হবে। একটা অ্যাপলিকেশনের ফন্টেন্ড দেখতে কেমন এটা যতটা গুরুত্বপূর্ণ, অ্যাপলিকেশনটা কতটা স্মৃথিলি পার্ফরম করছে সেটাও ঠিক ততটাই গুরুত্বপূর্ণ। একজন ফন্টেন্ড ডেভেলপার হিসেবে অ্যাপলিকেশনের পার্ফরমেন্স নিশ্চিত করা আপনার দায়িত্ব। আর এই গুরু দায়িত্ব সঠিক ভাবে পালন করতে হলে আপনার লাগবে অভিজ্ঞতা। ৩০ দিনে কমপ্লিট ফন্টেন্ড ডেভেলপমেন্ট এর মত চটকদার বিজ্ঞাপন দেখে উচ্ছ্বসিত হওয়ার কোন কারণ নেই। কারণ ভাল মাপের একজন ফন্টেন্ড ডেভেলপার হতে কম করে হলেও একবছরের বেশি সময় লাগবে। বড় ছোট মিলিয়ে ২৫-৩০ টা লাইব্রেরী সম্পর্কে জ্ঞান অর্জন করতে হবে। এছাড়াও ডেভেলপমেন্টের অসংখ্য টেকনিক শিখতে হবে। যা আমার মনে হয় না কোনভাবেই একবছরের পূর্বে ভালভাবে শেখা সম্ভব। যদি আপনার প্রোগ্রামিং এবং প্রৱ্রেম সল্লিং এর দক্ষতা অনেক বেশি থাকে সেই ক্ষেত্রে আপনার কিছুটা সময় কম লাগতে পারে। তারপরেও আমি মনে করি, এখানে সময় ব্যয় করা উচিত প্রয়োজনের থেকেও অনেক বেশি।

আমরা জানি ডেভেলপমেন্ট হচ্ছে সাহিত্য, আর এই সাহিত্য রচনার জন্য দরকার ভাষা। ডেভেলপমেন্টের জগতে এই একটা মাত্র ক্ষেত্রেই কোন অলটারনেটিভ নেই, কোন দ্বিমত নেই। একরকম জোর করেই যেন আমাদের ওপরে চাপিয়ে দেওয়া হচ্ছে জাভাস্ক্রিপ্ট নামক বিভীষিকাকে। আপনি যত বড় ডেভেলপারই হন না কেন, আপনি যদি ফন্টেন্ড ডেভেলপমেন্ট করতে চান, আপনাকে জাভাস্ক্রিপ্ট জানতেই হবে। ঠিক কতটা জানতে হবে? অনেকেই এই রকম প্রশ্ন হরহামেশায় করে থাকেন, যে ফন্টেন্ড ডেভেলপমেন্ট করতে কতটা জাভাস্ক্রিপ্ট জানতে হবে? জাভাস্ক্রিপ্ট হচ্ছে ভাষা, আর ডেভেলপমেন্ট হচ্ছে সাহিত্য। মনের মাধ্যম

---

মিশিয়ে সাহিত্য রচনার জন্য যতটা ভাষা জানা দরকার ঠিক ততটাই জাভাস্ক্রিপ্ট জানতে হবে।

জাভাস্ক্রিপ্ট হচ্ছে ফ্রন্টেন্ডের একমাত্র ল্যাংগুয়েজ যা ব্যবহার করে আমরা ওয়েব সাইটে বিভিন্ন ইন্টারেকশন, ডাইনামিক কন্টেন্ট এবং লজিক যুক্ত করতে পারি। জাভাস্ক্রিপ্ট খুবই ছোট একটা ল্যাংগুয়েজ। অনেকে তো এখনো এটাকে প্রোগ্রামিং ল্যাংগুয়েজই মনে করেনা, কারণ এর শেষে স্ক্রিপ্ট লেখা আছে। ছোট ল্যাংগুয়েজ হলেও এটা খুবই পাওয়ারফুল। এটা এতটাই পাওয়ারফুল যে অনেক গেম ইঞ্জিনেও এখন জাভাস্ক্রিপ্ট ব্যবহৃত হচ্ছে। জাভাস্ক্রিপ্টের বহুবিধ ব্যবহার নিয়ে আমরা পরবর্তীতে আলোচনা করবো। তবে প্রথমেই আমরা এটা মনে নিতে বাধ্য যে জাভাস্ক্রিপ্ট ছাড়া ফ্রন্টেন্ড আমাদের কোন গতি নেই।

জাভাস্ক্রিপ্টের প্রথম ব্যবহার হচ্ছে DOM Manipulation. HTML ব্যবহার করে একটা ওয়েবসাইট ডিজাইন করার পরে যদি আমরা সময়ের সাপেক্ষে সেখানে ডাইনামিক কোন কিছু যুক্ত করতে চাই, কোন বাটনে বা টেক্সটে ক্লিক হলে কিছু একটা করতে চাই অথবা নতুন নতুন এলিমেন্ট ক্রিয়েট বা রিমুভ করতে চাই, আমাদের প্রয়োজন পরবে জাভাস্ক্রিপ্ট। একটা HTML ডকুমেন্ট ব্রাউজার যখন লোড করে শো করে, তখন ওই HTML ডকুমেন্টের ভিতরে থাকা সমস্ত এলিমেন্ট একটা জাভাস্ক্রিপ্ট অবজেক্টে রূপান্তরিত হয়ে যায়। এই অবজেক্টের নাম ডকুমেন্ট অবজেক্ট বা Document Object Model (DOM). পরবর্তীতে এই জাভাস্ক্রিপ্ট অবজেক্টটা আমরা মনের মত পরিবর্তন বা পরিবর্ধন করতে পারি। যখন আমরা ডকুমেন্ট অবজেক্টে কোন পরিবর্তন করি, তখন ব্রাউজার সাথে সাথেই সেই পরিবর্তিত অবজেক্টটা আবার ব্রাউজারে রেন্ডার করে, যার ফলে আমরা আপডেটেড ইনফরমেশনগুলো দেখতে পাই। ডকুমেন্ট অবজেক্ট পরিবর্তন, পরিবর্ধন করার যেই কাজ জাভাস্ক্রিপ্ট ব্যবহার করে আমরা করে থাকি একেই বলে DOM Manipulation. একজন ফ্রন্টেন্ড ডেভেলপার হওয়ার পূর্ব শর্ত হচ্ছে জাভাস্ক্রিপ্ট ব্যবহার করে DOM Manipulation করতে পারা। এই ক্ষেত্রে DOM Manipulation করার অনেক লাইব্রেরী আপনি পাবেন। যার ভিতরে সব থেকে ভাল উদাহরণ হচ্ছে JQuery, কিন্তু আমি আপনাকে বলবো কোন রকম কোন লাইব্রেরী এর সাহায্য ছাড়াই আপনাকে DOM Manipulation শিখতে হবে। তাহলেই আপনি জাভাস্ক্রিপ্ট, DOM এবং ফ্রন্টেন্ড জাভাস্ক্রিপ্টের গুরুত্ব বুঝতে পারবেন যা আপনার পরবর্তীতে কাজে লাগবে।

---

DOM Manipulation শেখার পরেই আপনি যে কোন একটি লাইব্রেরী বা ফ্রেমওয়ার্ক বেছে নিতে পারেন। কিছু দিন পূর্বেও কখনো কল্পনা করি নি যে ফ্রন্টেন্ড ডেভেলপমেন্টের জন্য কোন লাইব্রেরী এর প্রয়োজন হবে। কিন্তু আজ ফ্রন্টেন্ড পুরো একটা আলাদা অ্যাপলিকেশন। আলাদা অ্যাপলিকেশন হিসেবে ডেভেলপড হচ্ছে, আলাদা অ্যাপলিকেশন হিসেবেই ডেপলয় বা হোষ্ট হচ্ছে। পুরোপুরি আলাদা ভাবে স্টেজিং, টেষ্টিং এবং বিউল্ড হচ্ছে। এর জন্য দরকার পড়ছে অসংখ্য টুলস যে গুলো ৫ বছর আগেও খুব একটা পরিচিত ছিল না। যেমন অ্যাপলিকেশন বিউল্ড করার জন্য দরকার বিউল্ড টুল, সব গুলো ফাইলকে বান্ডেল করার জন্য, কম্প্রেস করার জন্য দরকার বান্ডেলার, অ্যাপলিকেশন টেষ্ট করার জন্য দরকার ইউনিট টেষ্ট টুল, অটোমেটিক টেষ্ট এন্ড স্টেজিং টুল। আর এই বিষয় গুলো মোটেও সহজ কোন বিষয় না।

ফ্রন্টেন্ড ডেভেলপমেন্টের প্রসেসটাকে সহজ করে দিয়েছে NodeJS. NodeJS হচ্ছে একটা রানটাইম যা ব্যবহার করে আমরা জাভাস্ক্রিপ্টের কোডকে ব্রাউজারের বাইরে যেকোন জায়গায় রান করতে পারি। NodeJS এর সাথে আর একটা টুলস আছে যার নাম NPM বা Node Package Manager যা আমরা বহুবিধি কাজে ব্যবহার করছি। যেমন এটাই একটা বিউল্ড টুল হিসেবে কাজ করছে। নোড প্যাকেজ ম্যানেজারের কাছে আছে ৫ লক্ষেরও বেশি থার্ড পার্টি জাভাস্ক্রিপ্ট লাইব্রেরী যা আমরা টার্মিনাল বা কমান্ড লাইন ব্যবহার করে ছোট একটা কোড লিখেই ইন্সটল করে আমাদের প্রোজেক্টে ব্যবহার করতে পারি। নোডের কারণেই আমরা ফ্রন্টেন্ড Webpack এর মত বান্ডেলার ব্যবহার করতে পারি এবং JEST, Enzyme এর মত অটোমেটেড টেষ্ট টুল গুলোর ব্যবহার করতে পারি। এক কথায় বলতে গেলে ওয়েব ডিজাইন থেকে ফ্রন্টেন্ড ডেভেলপমেন্ট কনসেপ্টের জন্মই দিয়েছে নোডজেএস।

তবে ফ্রন্টেন্ড ডেভেলপার হিসেবে আমাদের নোডজেএস জানার প্রয়োজন নেই। আমাদের শুধু জানতে হবে NPM বা নোড প্যাকেজ ম্যানেজারটা কিভাবে কাজ করে। এর সাথে সাথে আর একটা টুলস সম্পর্কেও আমাদের অল্প একটু ধারণা থাকা দরকার আর তা হচ্ছে BabelJS. এটা একটা জাভাস্ক্রিপ্ট ট্রান্সপাইলার। আমরা আজকে যেই জাভাস্ক্রিপ্ট নিয়ে কাজ করি, কয়েক বছর পূর্বেও জাভাস্ক্রিপ্ট এতটা স্মার্ট ছিল না। এখন জাভাস্ক্রিপ্ট নতুন নতুন সব ফিচার, সিনট্যাক্স যুক্ত হয়েছে। জাভাস্ক্রিপ্টের এই নতুন ভার্সনকে বলা হয় ES6, যেই ভার্সনটা পুরাতন ব্রাউজারে ছিল না। তাই আপনার তৈরি করা অ্যাপলিকেশন যেন সব থেকে বেশি ব্রাউজারে সাপোর্ট দেওয়া যায় এই উদ্দেশ্যেই তৈরি করা হয়েছে BabelJS. যার কাজ, নতুন জাভাস্ক্রিপ্টের ফিচারকে, সিনট্যাক্সকে পুরাতন জাভাস্ক্রিপ্টের সিনট্যাক্সে রূপান্তর

---

করা। আমরা যখন কোড করবো তখন সব নতুন ফিচার ব্যবহার করবো, কিন্তু ব্রাউজার যখন এক্সিকিউট করবে সে সব পুরাতন ফিচারই দখতে পাবে। যার ফলে আমরা সব থেকে বেশি মানুষের কাছে আমাদের অ্যাপলিকেশন পৌঁছে দিতে পারব।

BabelJS এর সাথে সাথে আমরা Webpack ও ব্যবহার করে থাকি। Webpack হচ্ছে একটা বান্ডলার। এর কাজ অনেক গুলো জাভাস্ক্রিপ্ট ফাইলকে নিয়ে একটা মাত্র ফাইলে রূপান্তর করা। যখন আমরা কাজ করবো, তখন আমাদের কয়েকশ ফাইল নিয়ে কাজ করার প্রয়োজন হতে পারে। কিন্তু এত ফাইল বারবার লোড করতে বারবার নেটওয়ার্ক রিকুয়েস্টের দরকার পরে আর যেটা প্রচুর সময় নষ্ট করে। এই জন্য একটা বান্ডলার ব্যবহার করে সব গুলো ফাইলকে কমপ্রেস করে এক জায়গায় রেখে দেওয়া হয়। এতে সব গুলো ফাইল একটা ফাইলে রূপান্তরিত হয়ে যায়। এবং ওয়েব সাইট লোড নেওয়ার সময় একবারেই সমস্ত জাভাস্ক্রিপ্ট কোড ব্রাউজারের কাছে গিয়ে জমা হয়।

ফ্রন্টেন্ড ডেভেলপমেন্টের কথা বললে আরও দুইটা টুলসের নাম সামনে চলে আসে। একটা হচ্ছে লিন্টার আর দ্বিতীয়টা হচ্ছে কোড ফরমেটার। যে কোন ল্যাংগুয়েজে কোড করার একটা নির্দিষ্ট কনভেনশন আছে। আপনি যদি সেই কনভেনশন মনে কোড করেন তাহলেই সারা বিশ্বের কাছে আপনার কোড গ্রহণযোগ্যতা পাবে। আমরা মানুষ সব সময় ভুল করি, কোড করতে করতেও অনেক সময় এই কনভেনশন মনে চলা হয় না। যখনই আপনি এই রকম কনভেনশন ভুলে যাবেন, তখনই আপনাকে কনভেনশনের কথা মনে করিয়ে দেবে লিন্টার। সব থেকে জনপ্রিয় জাভাস্ক্রিপ্ট লিন্টার হচ্ছে ESLint। আর অটোমেটিক কোডের ফর্মেট ঠিক করার জন্য ব্যবহৃত হয় কোড ফর্মেটার। সব থেকে জনপ্রিয় কোড ফর্মেটার হচ্ছে Prettier.

এই যে টুলস গুলোর কথা বললাম এগুলো হচ্ছে সব থেকে বেশি ব্যবহৃত টুলস। এছাড়াও ডেভেলপমেন্ট প্রসেসটাকে সহজ করার জন্য আরও অসংখ্য টুলসের প্রয়োজন হতে পারে। টুলস গুলো নিজে নিজে কনফিগার করে ডেভেলপমেন্টের জন্য উপযোগী ইনভারমেন্ট তৈরি করতে শেখাটা খুবই গুরুতপূর্ণ। তবে বেশিরভাগ ক্ষেত্রেই আপনাকে এই কাজটা করতে হবে না। কারণ জনপ্রিয় সমস্ত ফ্রন্টেন্ড ফ্রেমওয়ার্ক বা লাইব্রেরী এর CLI বা কমান্ড লাইন টুল এবং Boilerplate সেটআপ আছে। এখানে শুধুমাত্র একটি কমান্ড রান করেই একটা প্রজেক্ট শুরু করা যায়।

---

ফ্রন্টেন্ডে কোন লাইব্রেরী বা ফ্রেমওয়ার্ক বেছে নিব এই নিয়ে প্রচুর দ্বিমত আছে। এক একজন একেকটা সাজেস্ট করে থাকে। মাত্র তিন চারটা পপুলার লাইব্রেরী, এর ভিতরে ডিসিশন নেওয়া খুব একটা কঠিন কিছু না। আবার কেউ যদি একটা ফ্রেমওয়ার্ক বা লাইব্রেরী ভালভাবে ব্যবহার করতে জানে, তাহলে কয়েকদিন সময় দিলে অন্য লাইব্রেরীও শিখে নিতে পারবে। বর্তমানে সব থেকে জনপ্রিয় লাইব্রেরী এবং ফ্রেমওয়ার্ক গুলো হচ্ছে, ReactJS, VueJS, Angular and Svelte. এই লিস্টে যে কোনো মুভৃত্তে পরিবর্তিত হতে পারে।

এই লিস্টে থাকা ReactJS সব থেকে ছোট এবং একমাত্র লাইব্রেরী। ছোট হলেও এটি ব্যবহার করে আপনি ফেসবুকের মত বড় বড় সব অ্যাপ্লিকেশন তৈরি করতে পারবেন। ReactJS তৈরিও করেছে ফেসবুকের টিম, ফেসবুক অ্যাপ্লিকেশনটার ফ্রন্টেন্ড ডেভেলপ করার জন্য। তাই আপনি নিশ্চিন্তেই ReactJS ব্যবহার করে যেকোনো সাইজের অ্যাপ্লিকেশন বানাতে পারেন। লিস্টে থাকা পরবর্তী দুইটাই হচ্ছে ফ্রেমওয়ার্ক। VueJS তৈরি করেছেন Evan You নামের এক ব্যক্তি আর Angular তৈরি করেছে টেক জায়ান্ট গুগল। জব মার্কেটে সব থেকে বেশি চাহিদা এখন অবধি ReactJS এর, তারপরেই অনেক ব্যবধানে আছে Angular এবং তার থেকে ব্যবধানে আছে VueJS. তবে জনপ্রিয়তার দিক থেকে VueJS এগিয়ে আছে সবার থেকে, এমনকি কিছু দিন আগে React কেও হার মানিয়েছে। VueJS and Angular যেহেতু ফ্রেমওয়ার্ক তাই অ্যাপ্লিকেশন ডেভেলপ করার সমস্ত টুলস এবং সাপোর্ট আপনি এদের কাছেই পাবেন। যদিও এই ক্ষেত্রে আপনার স্বাধীনতা ক্ষুণ্ণ হবে। যদি আপনি আমার মত স্বাধীন চেতা হয়ে থাকেন তাহলে আপনার জন্য React ই বেষ্ট হবে। আর আপনি যদি কমপ্লিট সল্যুশন পছন্দ করে থাকেন তাহলে আমি বলব Angular. তবে এইক্ষেত্রে আপনাকে Typescript শিখতে হবে। লিস্টে থাকা সর্বোশেষ টুলসটি একদমই নতুন কম্পিউটের এবং অন্ন কিছু দিনের ভিতরেই মানুষের মন জয় করে নিয়েছে। এটা না লাইব্রেরী না ফ্রেমওয়ার্ক, এটা একটা কম্পাইলার যা অনেকটা স্ট্যাটিক সাইট জেনারেটরের মত কাজ করে থাকে।

আপনি যেই লাইব্রেরী বা ফ্রেমওয়ার্কই বেছে নিন না কেন, প্রত্যেকেরই কিছু ভাল দিক এবং খারাপ দিক আছে, নিজস্ব লার্নিং কার্ড আছে, এবং এর সাথে জড়িত অসংখ্য থার্ড পার্টি টুলস আছে যার প্রায় সব কিছুই আপনাকে শিখতে হবে। আপনার ওয়েব ডিজাইনের যেই অভিজ্ঞতা আছে, সেই অভিজ্ঞতার খুব অন্ন কিছুই এখানে কাজে লাগবে। এখানে একটা সাইট, একটা কম্পানেন্ট ডিজাইনের নতুন কনসেপ্ট বিউল্ড হবে। কিন্তু সুখকর বিষয় এই যে, নতুন যেই কনসেপ্ট আপনার বিউল্ড হবে সেটা যেকোনো ফ্রন্টেন্ড ফ্রেমওয়ার্কের ক্ষেত্রেই

---

প্রযোজ্য হবে। প্রতিটা ফ্রন্টেন্ড ফ্রেমওয়ার্ক প্রায় কাছাকাছি ভাবেই কাজ করে থাকে একটা নির্দিষ্ট সমস্যার সমাধান করার জন্য। তাই যখন খুশি আপনি একটা থেকে আর একটাতে জাম্প করতে পারবেন, একই সময় একাধিক ফ্রেমওয়ার্ক নিয়েও কাজ করতে পারবেন। শুধু একবার আপনার মূল কনসেপ্টটা ক্লিয়ার হতে হবে।

এবার নিচয় বুঝতে পারছেন ওয়েব ডিজাইন এবং ফ্রন্টেন্ড ডেভেলপমেন্টের ভিতরে আসল পার্থক্যটা কোথায়? একটা ওয়েব অ্যাপ্লিকেশন তৈরির ক্ষেত্রে তার ফ্রন্টেন্ডটা বর্তমানে পুরোপুরি আলাদা ভাবে ডেভেলপ করা হয়। এর জন্য ডেভেলপমেন্ট টুলস এবং টেকনিকও ব্যবহার করা হয়। একটা মোবাইল অ্যাপ্লিকেশন আমরা যেভাবে ডেভেলপ করে থাকি, একটা ফ্রন্টেন্ড অ্যাপ্লিকেশনও আমরা প্রায় একই ভাবে তৈরি করে থাকি। একটা মোবাইল অ্যাপ্লিকেশনের জন্য আমাদের যেমন আলাদা করে ব্যাকেন্ড তৈরি করতে হয়, ঠিক একই ভাবে ফ্রন্টেন্ড অ্যাপ্লিকেশনের জন্যও আলাদা করে ব্যাকেন্ড অ্যাপ্লিকেশন তৈরি করতে হয়। মোবাইলের ক্ষেত্রে এমন অনেক অ্যাপ্লিকেশন সম্ভব যার কোনো ব্যাকেন্ডের দরকার নেই এবং এই রকম অ্যাপ্লিকেশন আমরা হরহামেশায় দেখে থাকি। একই ভাবে ব্যাকেন্ড বাদেই শুধু ফাংশনালিটি ব্যবহার করেই ফ্রন্টেন্ড অ্যাপ্লিকেশনও তৈরি করা সম্ভব।

ফ্রন্টেন্ড বর্তমানে একটা স্ট্যান্ডএলোন অ্যাপ্লিকেশন। সেই অ্যাপ্লিকেশন ডেভেলপ করার জন্য ডেভেলপমেন্টের সমস্ত রুলসই প্রয়োজন হয়। কিন্তু ওয়েবসাইট ডিজাইন মাটেও এই রকম কোনো বিষয় না। এর সাথে ডেভেলপমেন্টের কোনো সম্পর্ক নেই। একটা HTML, একটা CSS আর একটা Javascript ফাইল ক্রিয়েট করেই ওয়েবসাইট ডিজাইন করা যায়। বেশির ভাগ ওয়েব ডিজাইনই ব্যবহৃত হয় ব্যাকেন্ড অ্যাপ্লিকেশনের সাথে। তাহলে এবার আপনিই বলেন, কোনটাকে আমরা ডেভেলপমেন্ট বলবো আর কি কাজ করলে ফ্রন্টেন্ড ডেভেলপার বলা হবে।



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>



---

অধ্যায় ছয়

## সাহিত্যের ভাষা

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ জাভাস্ক্রিপ্ট প্রোগ্রামিং ল্যাংগুয়েজ
- ✓ জাভাস্ক্রিপ্ট এর মেইন কনসেপ্ট
- ✓ জাভাস্ক্রিপ্টে যা যা শিখতে হবে
- ✓ টাইপস্ক্রিপ্ট এ যা যা শিখতে হবে

---

ডেভেলপমেন্ট করতে হলে দরকার প্রোগ্রামিং ল্যাংগুয়েজ, আর ওয়েব ডেভেলপমেন্টের কথা আসলে দুইটা নাম সবার প্রথমেই শোনা যায়। একটি হচ্ছে জাভাস্ক্রিপ্ট, অন্যটি হচ্ছে পিএইচপি। বর্তমানে ওয়েবের আদি পিতা পিএইচপি কে বাদ দিয়েও ডেভেলপমেন্ট চলবে, কিন্তু জাভাস্ক্রিপ্টকে বাদ দিয়ে ওয়েবকে কল্পনায় করা যাবে না, অন্ততপক্ষে মডার্ন ওয়েবকে না। তবে জাভাস্ক্রিপ্টকে ধিরে নতুন প্রোগ্রামারদের ভিতরে রয়েছে প্রচুর কৌতুহল, প্রচুর ভয় এবং প্রচুর কনফিউশন। এর পিছনে সব থেকে বেশি দায়ী অধিশিক্ষিত কিছু ডেভেলপার যারা জাভাস্ক্রিপ্টের নামে প্রচুর গুজব ছড়িয়ে নতুনদের মনে ভয়ের সঞ্চার করেছে। আমি মানছি জাভাস্ক্রিপ্ট কিছুটা অন্তুত, তবে এর প্রতিটা অন্তুত সিনট্যাক্স এর পিছনে যুক্তি রয়েছে। হয়ত আমি জানিনা বা এখনো শিখিনি যে জাভাস্ক্রিপ্ট কম্পাইলার এই বিষয়টাকে কিভাবে ইন্টারপ্রেট করে। তাই বলে আমি কখনোই জাভাস্ক্রিপ্টের নামে গুজব ছড়ানোর অধিকার রাখি না। জাভাস্ক্রিপ্ট একটা প্রোগ্রামিং ল্যাংগুয়েজ। অন্য আর দশটা প্রোগ্রামিং ল্যাংগুয়েজের মত জাভাস্ক্রিপ্টেরও নিজস্ব সত্ত্ব থাকতেই পারে এবং সেই সত্ত্ব গুলো আমার কাছে কিছুটা অন্তুত লাগতেই পারে। এতে ভয় পাওয়ার মতো কোনো কারণ আমি দেখি না।

আমি অনেক গুলো প্রোগ্রামিং ল্যাংগুয়েজ নিয়ে কাজ করেছি। আমার কাছে মনে হয়, আমি যেই ল্যাংগুয়েজ গুলো নিয়ে কাজ করেছি তার মধ্যে সব থেকে ছোট ল্যাংগুয়েজটি হচ্ছে জাভাস্ক্রিপ্ট। আমি যদি লার্নিং কম্প্লেক্সিটির দিক থেকে বিবেচনা করি, তাহলে আমার হিসেব মতে জাভাস্ক্রিপ্ট জাভা এর থেকে ৫ ভাগের একভাগ এবং সি++ এর থেকে ৮ ভাগের একভাগ কঠিন। যদি টপিক্স বা ফিচার এর দিক থেকে চিন্তা করি তাহলে বলব, জাভা বা সি++ এর অর্ধেক বা তার কম। আমি জাভাস্ক্রিপ্ট শেখার মত তেমন কিছুই খুঁজে পাইনা। অল্প কয়েকটা ফিচার এমনভাবে ব্যবহার করা হয়েছে যেন একটা  $3 \times 3$  রুবিক্স কিউব। অল্প কয়েকটা ফিচার কিন্তু অসংখ্য ভাবে ব্যবহার করে সব ধরনের সমস্যার সমাধান করা যায় খুব কম কোড লিখে। তারপরেও আপনার যদি মনে হয় জাভাস্ক্রিপ্ট খুব কনফিউসিং তাহলে আমি বলব, এখনো জাভাস্ক্রিপ্ট এর সমস্ত বিষয় আপনি ভালো ভাবে একাপ্লেই করেন নি। আপনার কাছে যদি মনে হয় জাভাস্ক্রিপ্ট অনেক বড় ল্যাংগুয়েজ তাহলে আমার মনে হয় আপনি এখনো সি++, জাভা বা সিশার্প সমস্ত কনসেপ্ট দেখেন নি। সব গুলো ফিচার এবং জাভাস্ক্রিপ্ট ইঞ্জিন কিভাবে কাজ করে জানলে আপনি নিজেই নিজের ওপরে হাসবেন এবং মানতে বাধ্য হবেন যে জাভাস্ক্রিপ্ট খুব ছোট, সহজ এবং অ্যামেজিং ল্যাংগুয়েজ।

তবে জাভাস্ক্রিপ্টকে প্রথম ল্যাংগুয়েজ হিসেবে নেওয়াটা একটু বিস্তি। আমি ২০১২ সালে প্রথম প্রোগ্রামিং ল্যাংগুয়েজ হিসেবে জাভাস্ক্রিপ্ট শেখার চেষ্টা করেছিলাম Javascript Definitive

---

Guide বইটি পড়ে। কিছু দূর পড়ার পরে, অনেকটা চেষ্টা করার পরে, এখন আর আমি বলিনা যে জাভাস্ক্রিপ্ট আমার প্রথম ল্যাংগুয়েজ ছিল। যা দিয়ে আমি প্রোগ্রামিং শেখা শুরু করেছিলাম। তারপরেও আপনারা চেষ্টা করতেই পারেন। তবে এইক্ষেত্রে কিছু কিছু বিষয় ভালভাবে বুঝতে একটু সমস্যা হবে, কারণ জাভাস্ক্রিপ্ট একটা হাইলেভেল ল্যাংগুয়েজ। এটা Weakly Typed, মানে এখানে ডাইনামিক ভাবে কম্পাইলার ডাটা টাইপ বুঝে নেয়, ডেভেলপারকে বলে দিতে হয় না। এখানে তিনটি ভিন্ন ভিন্ন Programming Paradigm এ কোড করা যায় - Procedural, Object Oriented and Functional. আরও কনফিউশন তৈরি হয় যখন একই কোডে অবজেক্ট অরিয়েন্টেড এবং ফাংশনাল দুইটা প্রোগ্রামিং এর থিওরিই আঞ্চাই হয়। এরকম আরও কিছু কিছু বিষয় আছে যেগুলো একজন বিগিনার, যার প্রোগ্রামিং ল্যাংগুয়েজ নিয়ে কাজ করার পূর্ব অভিজ্ঞতা নেই তার জন্য বুঝে ওঠা খুব কঠিন হয়ে যাবে। আর যদি আপনি সিরিয়াস ভাবে ডেভেলপমেন্টকে আপনার ক্যারিয়ার হিসেবে নিয়ে থাকেন তাহলে তো আপনাকে ভাষা এবং ভাষার ব্যাকরণ শেখার পাঠ শেষ করেই এখানে আসতে হবে, সেই ক্ষেত্রে জাভাস্ক্রিপ্ট কখনোই আপনার প্রথম ল্যাংগুয়েজ হবে না।

পরবর্তীতে আমি যখন জাভাস্ক্রিপ্ট শুরু করি তখন আমার চার বছর জাভা নিয়ে কাজ করার অভিজ্ঞতা ছিল। জাভাস্ক্রিপ্টকে সিরিয়াস ভাবে নেওয়ার কোন উদ্দেশ্যও ছিল না, শুধুমাত্র ফ্রন্টেন্ড ডেভেলপমেন্টের জন্য এটা শিখতে চেয়েছিলাম, একটা অ্যাপলিকেশনের কাজে। তখন জানতামও না জাভাস্ক্রিপ্ট এত পাওয়ারফুল একটা ল্যাংগুয়েজ যা ব্যবহার করে অলমোচ্ন সবকিছুই ডেভেলপ করা যায়। আর দৃশ্য জন মানুষের মতই জাভাস্ক্রিপ্টকে তাচিল্যের চোখেই দেখতাম। এবং ভাবতাম এই রকম একটা স্ক্রিপ্টিং ল্যাংগুয়েজ শেখা তো কয়েক ঘণ্টার ব্যাপার। অবশ্যই, বেসিক শিখতে কয়েক ঘণ্টাও লাগবে না যদি কারোর জাভাতে চার বছরের অভিজ্ঞতা থাকে। কারণ জাভাস্ক্রিপ্টের বেশিরভাগ সিনট্যাক্স জাভা থেকেই নেওয়া হয়েছে। কিন্তু জাভাস্ক্রিপ্ট শুরু করার পরবর্তী দুই সপ্তাহ আমার সাথে যা ঘটলো তার জন্য আমি মোটেও প্রস্তুত ছিলাম না।

যতই ভাবি জাভাস্ক্রিপ্ট বুঝে গিয়েছি ততই নতুন নতুন সমস্যার সম্মুখীন হয়। জাভাস্ক্রিপ্টের পূর্বে ফাংশনাল প্রোগ্রামিং নিয়ে কাজ করার কোন অভিজ্ঞতা আমার ছিল না। কিন্তু জাভাস্ক্রিপ্ট শুরুর পূর্বে আমি কটলিন নিয়ে প্রচুর কাজ করেছিলাম। যার কারণে ফাংশনাল সিনট্যাক্স গুলো কিছুটা হলেও স্বস্তিদায়ক ছিল। তবে ফাংশনাল প্রোগ্রামিং এর বাস্তব ব্যবহার সম্পর্কে কোন কিছুই মাথায় আসছিলো না। তখন মাথায় যা কাজ করতো তার সবটাই অবজেক্ট অরিয়েন্টেড। যেকোন সমস্যা চোখের সামনে আসলেই কিভাবে ক্লাস ডায়াগ্রাম

---

বানাবো সেটা নিয়ে স্বপ্ন দেখতাম। আর জাভাস্ক্রিপ্ট এসে দেখি এখানে ক্লাস বলেই কিছু নেই। ফাংশন ব্যবহার করে কন্ট্রাক্টের আর প্রোটোটাইপ ব্যবহার করে ইনহেরিটেন্স করতে হয়। এগুলো যতটা না প্যারা দিয়েছে তার থেকে অনেক বেশি প্যারা দিয়েছে ফাংশনাল প্রোগ্রামিং এর ভিতরে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং। কোথায় this এর ভ্যালু কি সেটা বুঝতে পারাটায় তখন মনে হয়েছিল জীবনের একমাত্র লক্ষ্য। তারপরে শুনলাম ফাংশন নাকি অবজেক্ট। এটাও যে হতে পারে সেটা আমার এই ছেট্ট মস্তিষ্ক তখন কোন ভাবেই মনে নিতে পারছিলো না। সবসময় ফাংশনকে কল বা ইনভক করেই কাজ করেছি। এখানে এসে দেখি ফাংশনকে সরাসরি কল না করে তার ভিতরে থাকা মেথড দিয়েও কল করা যায়। আর জাভাস্ক্রিপ্ট বিগিনার হিসেবে তার প্রয়োজনীয়তা আমি কোনভাবেই উপলব্ধি করতে পারিনি তখন। লেক্সিক্যাল স্কোপ এবং ক্লোজারের কথা না বলাটায় শ্রেয়, কারণ এই শব্দ গুলো মনে রাখতেই দাঁত ভাঙ্গার উপক্রম হয়েছিল, কিভাবে কাজ করে, কি কাজ করে সেটা বোঝাতো বল্ল পরের বিষয়। Type Coercion এর জ্বালায় জাভাস্ক্রিপ্ট ছেড়ে দেওয়ার উপক্রম হয়েছিল। আসলে কখন যে কার ভ্যালু কি হবে সেটা প্রেডিক্ট করা মাত্র ২ সপ্তাহ জাভাস্ক্রিপ্ট প্রাকচিস করা নবজাতক জাভাস্ক্রিপ্ট প্রোগ্রামার হিসেবে আমার জন্য অসম্ভব ছিল। জাভাতে সব থেকে বেশি প্যারায় পড়তাম NullPointerException নিয়ে, আর জাভাস্ক্রিপ্টের অ্যাসিংক্রোনাস নেচারের জন্য এখানে হর হামেশায় undefined এর প্যারায় পরতে হতো এবং বুঝেও আসত না সব কিছু ঠিক ঠাক করার পরেও কেন undefined আসছে।

দুই সপ্তাহ এই রকম নানান সমস্যার সম্মুখীন হওয়ার পরে নিজের প্রতি বিত্তৃষ্ণা চলে আসল। মনে হল আমার সারা জীবনের পরিষ্পরাই বৃথা। কোন কিছুই আমি শিখিনি এত দিন। কোনো যোগ্যতা নেই আমার। তখন জাভাস্ক্রিপ্টের প্রতি ভয়, শ্রদ্ধা এবং সিরিয়াসনেস সবই বেড়ে গেল। বুঝতে পারলাম এভাবে হবে না, নতুন করে শুরু করতে হবে। এবার জাভাস্ক্রিপ্ট শুরু করার পূর্বে আমি যেই কাজটি করলাম আমার মনে হয় আপনারাও যদি এই কাজটি করেন তাহলে জাভাস্ক্রিপ্ট শিখতে একদমই সময় লাগবে না। সব সময় আমি যেকোনো কোড লিখতে গেলেই আমার মাদার ল্যাংগুয়েজ জাভাকে রেফারেন্স হিসেবে টানতাম, জাভাতে সমস্যাটা কিভাবে সমাধান করতাম সেই বিষয়টা নিয়ে ভাবতাম, এককথায় সব ল্যাংগুয়েজকেই জাভার সাথে গুলিয়ে কাজ করার চেষ্টা করতাম। কিন্তু এবার সিদ্ধান্ত নিলাম যে আমি জাভা ভুলে যাব। জাভা নামের কোনো ল্যাংগুয়েজ আমি কোনো দিন শিখি নি, আমি জানি না জাভা বলে কোনো প্রোগ্রামিং ল্যাংগুয়েজ আছে। আমি কোনো দিন কোনো প্রোগ্রামিং ল্যাংগুয়েজ শিখিও নি। শুধুমাত্র প্রোগ্রামিং এর ফান্ডামেন্টাল বিষয় গুলো বাদে সব

---

কিছুকে ভুলে যাওয়ার চেষ্টা করলাম এবং একটা ফ্রেস মাইন্ডে জাভাস্ক্রিপ্ট শেখা শুরু করলাম।

জাভাস্ক্রিপ্ট অন্তর্ভুক্ত কিন্তু জাভাস্ক্রিপ্ট সুন্দর, জাভাস্ক্রিপ্ট ডাইনামিক কিন্তু এখানে আগে থেকে কিছু প্রেভিউ করা যায় না কথাটা শতভাগ সত্যি নয়। জাভাস্ক্রিপ্ট একটা অন্যরকম ক্রিয়েশন। আপনি যদি জাভাস্ক্রিপ্ট শিখতে চান আপনাকে জাভাস্ক্রিপ্টের মত করে প্রোগ্রামিং কে দেখতে হবে। জাভাস্ক্রিপ্টের কম্পাইলারের মত করে আপনাকে চিন্তা করতে হবে। জাভাস্ক্রিপ্ট ছোট্ট একটা ল্যাংগুয়েজ, অন্তর্বর্তী কয়েকটা ফিচারকেই এমনভাবে বার বার ব্যবহার করা হয়েছে, মনে হয় যেন একটা পার্জেল সন্তুষ্ট করতে দেওয়া হয়েছে। যখন আমি জাভাস্ক্রিপ্টকে তার নিজের মত করে দেখার চেষ্টা করলাম খুব বেশি সময় আমার লাগে নি একে বুঝতে। ১ মাসের ভিতরেই প্রায় এর সমস্ত মূখ্য বিষয় গুলো আয়ত্ত করা হয়ে গিয়েছিল।

পৃথিবীতে অসংখ্য প্রোগ্রামিং ল্যাংগুয়েজ আছে। আমরা সবাই বলি, একটা ল্যাংগুয়েজ থেকে আর একটা ল্যাংগুয়েজে সুইচ করতে খুব বেশি ঝামেলা হয় না। কথাটা সত্য, তবে আর একটা কথাও সত্য যে প্রতিটা ল্যাংগুয়েজেরই নিজস্ব সত্ত্ব আছে। একটা ল্যাংগুয়েজে কয়েকটা সিনট্যাক্সি লিখে সেই ল্যাংগুয়েজের সমস্ত সত্ত্ব সম্পর্কে ধারণা অর্জন করা সম্ভব না। এর জন্য দরকার সময় এবং অভিজ্ঞতা। আর এই অভিজ্ঞতা আসবে সমস্যা থেকে। আপনি যত বেশি গভীরে যাবেন তত বেশি পরিমাণে সমস্যার সম্মুখীন হবেন, আর যত বেশি সমস্যার সম্মুখীন হবেন তত বেশি নতুন নতুন দুয়ার আপনার সামনে উন্মুক্ত হতে থাকবে। এটা একটা অন্যরকমের ত্রুটি, যেই ত্রুটির নেশা আপনাকে আস্তে আস্তে গ্রাস করে ফেলবে, আর আপনি নেশায় নেশা গ্রস্ত হতে হতে জড়িয়ে পরবেন টেকনোলজির প্রেমে।

জাভাস্ক্রিপ্ট এত সুন্দর, এত আকর্ষণীয় প্রোগ্রামিং ল্যাংগুয়েজ হওয়ার পরেও আমরা অনেকেই জাভাস্ক্রিপ্টকে দাম দিতে চাই না। এখনো জাভাস্ক্রিপ্টকে প্রোগ্রামিং ল্যাংগুয়েজ ভাবতে অনেকেই নারাজ। তবে আমি তাদেরকে দোষ দিতে চাই না। ১৯৯৫ সালে যখন জাভাস্ক্রিপ্ট এসেছিল তখন তার কাজ ছিল ওয়েব পেজে কিছুটা ইন্টার্যাকটিভিটি যুক্ত করা। প্রায় ১৫ বছর ধরেই সে এই একটা মাত্র কাজই করে আসছিল। ২০০৯ সালে প্রথম জাভাস্ক্রিপ্ট নিয়ে অন্যরকম স্বপ্ন দেখেন Ryan Dahl নামক একজন ডেভেলপার। তিনিই প্রথম চিন্তা করেন এত সুন্দর একটা ভাষাকে কেন আমরা অন্য কোন কাজে ব্যবহার করছি না? তাই তিনি তখনই NodeJS নিয়ে কাজ শুরু করেন এবং ২০১১ সাল থেকে মানুষ জাভাস্ক্রিপ্টের আসল সৌন্দর্য, আসল ক্ষমতা বুঝতে শুরু করে NodeJS এর বদৌলতে।

---

কিন্তু আমাদের দেশে জাভাস্ক্রিপ্ট বিস্তারের এই খবরটা এখনো খুব একটা ছড়ায় নি। এখনো আমরা সেই আদিম যুগেই পরে আছি। খুব অল্প কিছু সংখ্যক মানুষ যারা গ্লোবাল ট্রেন্ড ফলো করে তারাই শুধুমাত্র জাভাস্ক্রিপ্ট এবং নোডজেএস এর ক্ষমতা সম্পর্কে অবগত হয়েছেন এবং এগুলো নিয়ে কাজ করছেন। বাকিদের কাছে হয়ত খবরটি এখনো পৌঁছায়নি যে, জাভাস্ক্রিপ্ট ব্যবহার করে এখন পিএইচপি এর থেকেও অনেক পাওয়ারফুল অ্যাপ্লিকেশন তৈরি করা যায় আরও সহজে, আরও কম সময়ে।

কয়েকবছর আগে যদি আমরা ডেস্কটপ অ্যাপ্লিকেশন বানানোর কথা চিন্তা করতাম তাহলে মাথায় আসত C++, C#, Java এর মত ল্যাংগুয়েজ গুলোর নাম। যদি অ্যান্ড্রয়েড অ্যাপ্লিকেশন বানাতে চাইতাম তাহলে Java or C++, iOS or Mac অ্যাপ্লিকেশনের জন্য Objective C or Swift, ওয়েব ডেভেলপমেন্টের কথা আসলেই মাথায় আসত Php, Asp, Java EE, Ruby অথবা Python এর মত ল্যাংগুয়েজ গুলোর নাম। যদি ডাটা সাইন্সের কথা বলতাম তাহলে চোখ বন্ধ করে Python or R এর কথাই সবার মনে পড়ত। গেম ডেভেলপমেন্ট করতে চাইলে C++, C# or Java ব্যবহার করতাম। কিন্তু আজকে এই সব কাজ আপনি করতে পারবেন শুধুমাত্র একটা ল্যাংগুয়েজ ব্যবহার করে, আর সেটা হচ্ছে জাভাস্ক্রিপ্ট। শুধু করতে পারবেন এমনটা ভাবার দরকার নেই, সফল ভাবে করতে পারবেন। ডেভেলপমেন্ট জগতে একটা নতুন মোড়ের জন্ম দিয়েছে এই জাভাস্ক্রিপ্ট। ডেস্কটপ থেকে মোবাইল অ্যাপ্লিকেশন, ওয়েব ডেভেলপমেন্ট থেকে গেম ডেভেলপমেন্ট, Embeded System থেকে IOT সব জায়গায় আপনি সফলভাবে নিশ্চিন্তে জাভাস্ক্রিপ্ট ব্যবহার করে ডেভেলপমেন্ট এর কাজ সম্পন্ন করতে পারেন।

আজকে ওপেন সোর্স যেই কোড এডিটর ছাড়া ডেভেলপারদের চলেই না, VSCode, এই এডিটরটি তৈরি করা হয়েছে জাভাস্ক্রিপ্টের সুপারসেট টাইপস্ক্রিপ্ট ব্যবহার করে, যা পরবর্তীতে জাভাস্ক্রিপ্টই কনভার্ট হয়ে যায়। মানে আপনি বলতে পারেন এটা জাভাস্ক্রিপ্ট পাওয়ারড একটা টেক্নিক এডিটর, যা ক্রস প্লাটফর্ম একটা অ্যাপ্লিকেশন এবং এটা আমরা সমস্ত অপারেটিং সিস্টেমেই কোনো সমস্যা ছাড়াই ব্যবহার করছি। জাভাস্ক্রিপ্ট ব্যবহার করে আপনি যেকোনো ক্রস প্লাটফর্ম মোবাইল বা ডেস্কটপ অ্যাপ্লিকেশন বানাতে পারেন কোনো রকম কোনো ন্যাটিভ ল্যাংগুয়েজের সাহায্য ছাড়াই এবং এই সিস্টেম এখন বিশ্ববাজারে খুবই জনপ্রিয়। ওয়েব ডেভেলপমেন্টের কথা যদি বলি, বর্তমানে ডাটা ড্রাইভেন অ্যাপ্লিকেশনের চাহিদা ব্যাপক। বলতে পারেন নতুন সমস্ত অ্যাপ্লিকেশনই এই ডাটা ড্রাইভেন ওয়েবে তৈরি করা হচ্ছে। আর এই ধরনের অ্যাপ্লিকেশন তৈরির ক্ষেত্রে জাভাস্ক্রিপ্টের কোনো জুড়ি নেই।

---

API Driven যে কোনো অ্যাপলিকেশন ডেভেলপমেন্টের ক্ষেত্রে ডেভেলপারদের প্রথম চয়েজ নোডজেএস বা জাভাস্ক্রিপ্ট। মেশিন লার্নিং বা ডাটা সাইন্সের মতো কাজ গুলো জাভাস্ক্রিপ্ট ব্যবহার করে করার জন্য এগিয়ে আসছে গুগলের মতো কোম্পানি গুলো। বিশ্বের প্রথম দশটা টেক কোম্পানির ভিতরে আপনি একটা কোম্পানিকেও দেখাতে পারবেন না যারা জাভাস্ক্রিপ্টের মাল্টিপ্ল ব্যবহার করছে না। Unity এর মত জনপ্রিয় গেম ইঞ্জিনও জাভাস্ক্রিপ্টকে সাপোর্ট করা শুরু করেছে অন্ন কিছু পরিবর্তন করে। এছাড়াও জাভাস্ক্রিপ্টের জন্য অসংখ্য ওপেন সোর্স গেম ইঞ্জিন রয়েছে। এত বড় তালিকার ভিতরে ফ্রন্টেন্ড ডেভেলপমেন্টের কথা না হয় আর নাই বলি। কারণ এই কাজের জন্যই তো জাভাস্ক্রিপ্টের জন্ম।

জাভাস্ক্রিপ্টকে ছোটোখাটো একটা স্ক্রিপ্টিং ল্যাংগুয়েজ ভেবে ভুল করার সময় আর নেই। আমরা যদি এই ভুলটা এখনো করতে থাকি তাহলে আসলেই বিরাট বড় ভুল হয়ে যাবে। পৃথিবী যে দিকে এগিয়ে যাচ্ছে আমরা তার সাথে আর তাল মিলিয়ে এগিয়ে যেতে পারবো না, এমনিতেই আমরা অনেক পিছিয়ে আছি। একমাত্র জাভাস্ক্রিপ্ট শিখে যদি অনেক গুলো সাইটে ডেভেলপমেন্ট করা যায় তাহলে একজন ডেভেলপার হিসেবে কেন আমি এই গোল্ডেন অপচুনিটিকে হারাবো? জাভাস্ক্রিপ্টতো খুব কঠিন কোন ল্যাংগুয়েজ না। জাভাস্ক্রিপ্টের বর্তমান ইকো সিস্টেম যে কোন ল্যাংগুয়েজ এর ইকো সিস্টেমের থেকে অনেক অনেক বড়। বিশাল বড় তার কমিউনিটি। আপনি যদি একবার এই বিশাল কমিউনিটির ভিড়ে জাভাস্ক্রিপ্ট ইকো সিস্টেমে ঢুকে পরতে পারেন, তাহলে আপনি আর বের হয়ে যেতে চাইবেন না।

জাভাস্ক্রিপ্টের ইকো সিস্টেমটা গড়ে উঠেছে নোডকে কেন্দ্র করে। তারমানে এই না যে আপনাকে নোডজেএস ও জানতে হবে। আপনাকে শুধু জানতে হবে নোড প্যাকেজ ম্যানেজার, যা নোডজেএস এর সাথে অটোমেটিক্যালি ইন্সটল হয়ে যায় এবং টার্মিনাল বা কমান্ড লাইনের মাধ্যমে একে ব্যবহার করতে হয়। খুব অন্ন কিছু কমান্ড আছে এটি অপারেট করার জন্য। এই একটা টুলস আপনার সামনে সন্তানার দুয়ার খুলে দিবে। অ্যাপলিকেশন বিউল্ড করা, টেষ্ট করা, ডেপলয় করা, ডেভেলপমেন্ট সার্ভার, লক্ষ লক্ষ থার্ড পার্টি লাইব্রেরী সব আপনার হাতের মুঠোয় চলে আসবে শুধু এই একটি মাত্র টুলসের কারণে। নোডজেএস এর জনপ্রিয়তার পিছনের সব থেকে বড় রহস্য হচ্ছে এই নোড প্যাকেজ ম্যানেজার। আপনি যখন জাভাস্ক্রিপ্ট শিখবেন অথবা জাভাস্ক্রিপ্ট ব্যবহার করে যে কোন প্লাটফর্মে কাজ করবেন তখন আপনার সবথেকে প্রয়োজনীয় টুলস হবে এই নোডজেএস এবং নোড প্যাকেজ ম্যানেজার।

২০১৫ সালে জাভাস্ক্রিপ্টের একটা যুগান্তকারী আপডেট আসে যা ES6 বা ES2015 নামে পরিচিত। এর পূর্বে জাভাস্ক্রিপ্ট আসলেই একটা খেলনা ল্যাংগুয়েজ ছিল। প্রয়োজনীয় অনেক ফিচারই ছিল না। এখন যেইসব ফিচারের জন্য ডেভেলপাররা জাভাস্ক্রিপ্টকে এত ভালোবাসে তার কিছুই ছিল না এর আগে। তবে নতুন নতুন ফিচার আসলেও জাভাস্ক্রিপ্টের কম্পাইলারে কিন্তু কোনো পরিবর্তন আসে নি, শুধুমাত্র একটা মুখোশ পড়িয়ে দিয়েছে কিছু নতুন নতুন সিনট্যাক্স এর মাধ্যমে। নতুন যারা জাভাস্ক্রিপ্ট শিখতে আসে তাদের অনেকের মনেই এই প্রশ্নটা ঘূরপাক থায় যে আমাকে সেই ওল্ড ব্যাড লুকিং জাভাস্ক্রিপ্টটা শিখতে হবে কিনা? অবশ্যই শিখতে হবে, তা না হলে কোন ফিচারটা কিভাবে কাজ করছে তার সবটাই আপনার অজানা থেকে যাবে। কারণ এখনো বিহান্ত দ্যা সিন ওই ব্যাড লুকিং ওল্ড জাভাস্ক্রিপ্ট কোড গুলোই এক্সিকিউট হচ্ছে। শুধু আমরা কিছু সহজ সুন্দর সিনট্যাক্স পেয়েছি যা আমাদের জীবনটাকে অনেক সহজ করে দিয়েছে।

এই হিসেবে ধরলে জাভাস্ক্রিপ্টের দুইটা ভার্সন। আর একজন ভালো জাভাস্ক্রিপ্ট প্রোগ্রামার হতে হলে দুইটা ভার্সন সম্পর্কেই পূর্ণ জ্ঞান রাখতে হবে। আপনি যদি ওল্ড ভার্সন যাকে আমরা ES5 বলে থাকি, সেটা ভালোভাবে শিখতে পারেন তাহলে ES6 এ এসে আপনাকে শুধু সহজ কিছু সিনট্যাক্স শিখতে হবে। প্রোগ্রামিং এর ফান্ডামেন্টালটা জাভাস্ক্রিপ্ট একবার ঝালাই করে নিয়েই আপনি এর ফাংশনাল অংশটা নিয়ে কাজ শুরু করতে পারেন। এখানে আপনি শিখবেন জাভাস্ক্রিপ্টে ফাংশন কিভাবে কাজ করে, কিভাবে ফাংশনকে একটা সাধারণ ভ্যালু হিসেবে ব্যবহার করা যায়, কিভাবে একটা ফাংশনের ভিতরে ফাংশন পাস করে স্থান থেকে আর একটা নতুন ফাংশন রিটার্ন করা যায়। এর সাথে সাথে ফাংশনের স্কোপ, লেক্সিক্যাল স্কোপ, ক্লোজার সম্পর্কে অল্প কিছু ধারণা নিয়ে নেওয়া যেতেই পারে। যখন আপনি এই বিষয় ভালোভাবে বুঝতে চেষ্টা করবেন তখন দেখবেন সব কিছুই কেমন জানি ঝাপসা ঝাপসা লাগছে। এই বিষয়কে পরিষ্কার করার জন্য এক্সিকিউশন কন্ট্রুক্ট সম্পর্কে বিস্তারিত জেনে নিতে হবে। পিউর ফাংশন, ফার্স্ট ক্লাস ফাংশন, হায়ার অর্ডার ফাংশন, কলব্যাক ফাংশনের মত কনসেপ্ট গুলোকে ভালোভাবে ঝালাই করে নিতে হবে। কারণ জাভাস্ক্রিপ্ট সব সময় আপনি এই বিষয় গুলো নিয়েই কাজ করবেন।

ফাংশনের কনসেপ্ট গুলো ভালোভাবে পরিষ্কার হয়ে গেলেই আপনি জাভাস্ক্রিপ্টের অবজেক্ট অরিয়েন্টেড সেকশনে জাম্প করবেন। জাভাস্ক্রিপ্ট মাটেও কোন পিউর অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ না এবং এখানে অবজেক্ট অরিয়েন্টেডের সমস্ত ফিচারও নেই। ফাংশন এবং প্রোটোটাইপ ব্যবহার করে এখানে অবজেক্ট অরিয়েন্টেড ইমপ্লিমেন্ট করা হয়েছে। তাই অবজেক্ট

---

অরিয়েন্টেড সম্পর্কে ভাল ধারণা না থাকলে বিষয় গুলো মাথার ওপর দিয়ে যাওয়ার সম্ভাবনাই বেশি। ES6 এ আপনারা অবজেক্ট অরিয়েন্টেডের দুই একটা ফিচার দখলেন যেমন class, inheritance কিন্তু এই গুলো সবই সিন্ট্যাক্টিক সুগার, মানে এক্সিকিউট হয়ে প্রোটোটাইপের কাছেই চলেই যাবে। এখানে খুব ভালোভাবে জাভাস্ক্রিপ্টের অবজেক্ট সম্পর্কে ধারণা অর্জন করতে হবে। কিভাবে ফাংশন ব্যবহার করে কনষ্ট্রাক্টর বা ফ্যাক্টরি প্যাটার্ন ইমপ্লিমেন্ট করতে হয় সেটা বুঝতে হবে। প্রোটোটাইপ অবজেক্ট, প্রোটোটাইপিক্যাল ইনহেরিটেন্স, গ্লোবাল অবজেক্ট এবং this সম্পর্কে ভাল একটা ধারণা অর্জন করতে হবে। এক্সিকিউশন কন্টেক্টের ওপরে ভিত্তি করে কিভাবে this এর ভ্যালু পরিবর্তিত হয়, কিভাবে আপনি এক্সিকিউশন কন্টেক্টকে পরিবর্তিত করতে পারেন, এই সব বিষয় গুলো ভালো ভাবে বুঝতে হবে জাভাস্ক্রিপ্ট অবজেক্ট অরিয়েন্টেড নিয়ে কাজ করতে চাইলে।

এর পরে আপনি জাভাস্ক্রিপ্টের অ্যাসিংক্রোনাস নেচার নিয়ে পড়াশোনা করতে পারেন, তবে এই ক্ষেত্রে আপনার জাভাস্ক্রিপ্ট ইঞ্জিন সম্পর্কে ধারণার দরকার পড়বে। কারণ অ্যাসিংক্রোনাস বিহেভিয়ারটা আসলে ল্যাংগুয়েজের কোন বিষয় না। একটা কোড আমাদের কম্পাইলার কিভাবে এক্সিকিউট করছে সেটা যদি আমরা বুঝতে পারি তাহলেই আমরা জাভাস্ক্রিপ্টের এই বিহেভিয়ারটা সম্পর্কে অবগত হতে পারব। জাভাস্ক্রিপ্ট ইঞ্জিন নিয়ে পড়াশোনা করতে করতে আপনি জানবেন জাভাস্ক্রিপ্টের পিছনে একটি JIT (Just In Time) কম্পাইলার কাজ করছে। আপনি এও জানবেন জাভাস্ক্রিপ্ট একই সাথে কম্পাইল এবং ইন্টারপ্রেটেড ল্যাংগুয়েজ। আপনাকেই খুঁজে বের করতে হবে যে জাভাস্ক্রিপ্টের এইরকম ন্যাচারের পিছনের কারণটা কি?

এর পরে আপনি ES6 এর সিন্ট্যাক্ট গুলো শিখে নিতে পারেন। ES6 যখন আপনি শিখবেন তখন আরও কিছু টুলস যেমন BabelJS, Webpack, ESLint, Live Server এর মত অসংখ্য কিছু টুলস আপনার টুলবক্সে যুক্ত হবে। এখন আপনি জাভাস্ক্রিপ্ট ব্যবহার করে কিছু প্রোগ্রাম সল্লু করবেন। HackerRank, SPOJ এর মত সাইট গুলোতে জাভাস্ক্রিপ্ট ব্যবহার করে প্রোগ্রাম সল্লু করতে পারেন। আবার DOM ম্যানিপুলেশন শিখে বাস্তব জীবনের বিভিন্ন অ্যাপলিকেশন তৈরি করার মাধ্যমেও জাভাস্ক্রিপ্টের জ্ঞান বৃদ্ধি করতে পারেন। মজা করার জন্য দুই একটা গুগল ক্রোম এর এক্সটেনশন বা অ্যাপলিকেশনও বানিয়ে দেখতে পারেন। আসল কথা হচ্ছে আপনাকে জাভাস্ক্রিপ্ট ল্যাংগুয়েজটা অনুশীলন করতে হবে। সেটা যেভাবেই হোক না কেন।

## প্রথম ল্যাংগুয়েজ হিসেবে জাভাস্ক্রিপ্ট

অনেকেই জাভাস্ক্রিপ্টকে প্রথম ল্যাংগুয়েজ হিসেবেই বেছে নেয় শর্টকাটে ডেভেলপার হওয়ার জন্য। কিন্তু কিছু দিন পরেই তারা বিভিন্ন সমস্যার সম্মুখীন হয়। আমি কখনোই কাউকে প্রথম ল্যাংগুয়েজ হিসেবে জাভাস্ক্রিপ্টকে বেছে নেওয়ার পরামর্শ দেই না। অবশ্যই আপনি চাইলে জাভাস্ক্রিপ্ট দিয়ে শুরু করেই আপনার প্রোগ্রামিং ক্যারিয়ার অনেক ভালো করতে পারবেন। কিন্তু সেই ক্ষেত্রে চলার পথটা একটু বেশিই কাটাময় হওয়ার সম্ভাবনা বেশি থাকে।

### জাভাস্ক্রিপ্ট যা যা শিখতে হবে

- Programming Fundamentals
- Data Types and Type Coercion
- Conditions, Loops and Basic Function
- Arrays and Objects
- Basic Functional Programming Concepts
- Global and Lexical Scope
- Execution Context
- Variable Object and Hoisting
- Scope Chain and Closure
- Object Oriented Theory
- Object Creation Patterns
- Prototype and Inheritance
- Global Object & this
- Bind, Call & Apply
- Object and Object References
- Asynchronous Programming
- Error Handling
- ECMA Script 6
- Overview of Javascript Engine

**নোটঃ** জাভাস্ক্রিপ্ট শেখার সময় জাভাস্ক্রিপ্টকে কখনোই ফ্রন্টেন্ড ল্যাংগুয়েজ হিসেবে শিখবেন না। একে অন্যান্য প্রোগ্রামিং ল্যাংগুয়েজের মতই গুরুত্ব সহকারে এবং অ্যাবস্ট্রাক্ট ভাবে শিখবেন যা আপনার পরবর্তীতে নোডজেএস শেখার সময় অনেক কাজে দিবে।

# জাভাস্ক্রিপ্ট শিখার রেফারেন্স

## Must Read Javascript Books:

- Eloquent Javascript: A Modern Introduction to Programming By Marjin Heijer
- Javascript: The Definitive Guide By David Flanagan
- Effective Javascript By David Herman
- You Don't Know JS By Kyle Simpson
- Javascript: The Good Parts By Douglas Crockford

## Youtube Channels to Learn Javascript

- [Learn with Hasin Haydar](#) (Bangla)
- [JS Bangladesh](#) (Bangla)
- [Web Developer BD](#) (Bangla)
- [Stack Learner](#) (Bangla)
- [Traversy Media](#) (English)
- [Academind](#) (English)
- [Wes Bos](#) (English)
- [Fun Fun Function](#) (English)

## Websites to Learn Javascript

- [Javascript.info](#)
- [Mozilla Developer Networks](#)
- [Zonayed JS](#)
- [Free Code Camp](#)
- [Geeks For Geeks](#)

## Training Program for Javascript

- [Stack Learner Premium Courses](#)
- [Stack Learner Offline Bootcamps](#)

---

জাভাস্ক্রিপ্ট আজকে অনেক পাওয়ারফুল, প্রচুর পাওয়ারফুল সিনট্যাক্স আছে এর মূলিতে। কয়েক বছর আগে কিন্তু জাভাস্ক্রিপ্ট এতটা পাওয়ারফুল ছিল না। আজকে জাভাস্ক্রিপ্ট পাওয়ারফুল হলেও কিছু কিছু ক্ষেত্রে এর এখনো অনেক ল্যাক রয়েছে, যেমন অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং। জাভাস্ক্রিপ্ট আপনি আজও পূর্ণ অবজেক্ট অরিয়েন্টেড এর স্বাদটা পাবেন না। এরপরে, জাভাস্ক্রিপ্ট কোনো টাইপড ল্যাংগুয়েজ না, মানে এখানে কখনোই আপনাকে ডাটা টাইপ বলে দিতে হয় না। এর ফলে অ্যাপ্লিকেশনে অনেক রকম ইরোর সৃষ্টি হওয়ার সম্ভাবনা থেকে যায়। যারা বড় বড় ডেভেলপার আছে, যাদের মূলিতে C++, C#, Java এর মত ল্যাংগুয়েজে কাজ করার অভিজ্ঞতা রয়েছে তারা সহজেই জাভাস্ক্রিপ্টের মত ল্যাংগুয়েজেকে গ্রহণ করে নিতে পারে না। আমার নিজেরও অনেক সমস্যা হয়েছিল যেহেতু আমি জাভা ব্যাকগ্রাউন্ডের ছিলাম। আপনাদের জন্য সুখবর হচ্ছে, আপনাদেরকে নিয়ে চিন্তা করেছে মাইক্রোসফট এবং আমাদেরকে উপহার দিয়েছে টাইপস্ক্রিপ্ট (TypeScript)। আপনি যদি টাইপস্ক্রিপ্ট কোড করেন তাহলে বুঝতেই পারবেন না যে জাভাস্ক্রিপ্ট কোড করছেন।

টাইপস্ক্রিপ্ট হচ্ছে জাভাস্ক্রিপ্টের একটা ট্রান্সপাইল্ড ল্যাংগুয়েজ। এখানে আপনি মজার মজার সব ফিচার পাবেন যা জাভাস্ক্রিপ্ট নেই, সাথে জাভাস্ক্রিপ্টে যে গুলো আছে সেগুলোতো পাবেনই। সেই হিসেবে বলা যায় টাইপস্ক্রিপ্ট হচ্ছে জাভাস্ক্রিপ্টের সুপারসেট। এর প্রধান বিশেষত্ব হচ্ছে স্ট্রাংলি টাইপড। জাভাস্ক্রিপ্টে টাইপ সিস্টেমের যেই সমস্যাটা ছিল টাইপস্ক্রিপ্ট সেই সমস্যাটার একটা পাকাপোক্ত সমাধান করেছে। এর সাথে সাথে আপনি পাবেন ফিচার রিচ অবজেক্ট অরিয়েন্টেড সিনট্যাক্স। আপনি যদি জাভা বা সি শার্প থেকে আসেন তাহলে বুঝতেই পারবেন না আপনি অন্য ল্যাংগুয়েজে কাজ করছেন।

টাইপস্ক্রিপ্ট খুবই জনপ্রিয় একটা ল্যাংগুয়েজ। আপনি যদি জাভাস্ক্রিপ্টের দুনিয়াতে এসেও আপনার পুরাতন স্টাইলে, সেফ কোড লিখতে চান তাহলে টাইপস্ক্রিপ্টের কোন জুড়ি নেই। নিঃসন্দেহে এটা শিখে নিতে পারেন এবং অফিসিয়াল ডকুমেন্টেশন দেখে শিখতে সর্বোচ্চ একদিন সময় লাগতে পারে। তবে জাভাস্ক্রিপ্ট না শিখেই টাইপস্ক্রিপ্ট শিখতে চাওয়ার মত বোকামি আর কোনো কিছুতেই হবে না। আপনি জাভাস্ক্রিপ্ট শেখার পরেই টাইপস্ক্রিপ্ট শিখতে পারবেন এবং এটা শিখবেন শুধুমাত্র সেফ কোড লেখার জন্য।

## টাইপস্ক্রিপ্ট শিখা কর্তা জরুরি

বর্তমান সময়ের জন্য টাইপস্ক্রিপ্ট শিখা খুবই জরুরি। কারণ সফটওয়্যার ইন্ডাস্ট্রিগুলো কোর জাভাস্ক্রিপ্টের থেকে টাইপস্ক্রিপ্টের ওপরেই বেশি ভরসা করছে। তাই ফ্রন্টেন্ড, ব্যাকেন্ড মোবাইল ডেভেলপমেন্ট সব জায়গাতেই টাইপস্ক্রিপ্টের ব্যবহার দখন যাচ্ছে। থার্ড পার্টি লাইব্রেরী ব্যবহারের ক্ষেত্রেও আমরা এখন টাইপস্ক্রিপ্টের সাপোর্ট পাচ্ছি। বড় বড় সমস্ত লাইব্রেরী বা ফ্রেমওয়ার্ক গুলোও তৈরি হচ্ছে টাইপস্ক্রিপ্ট ব্যবহার করেই। তাই আমার মনে হয়, টাইপস্ক্রিপ্ট শিখে রাখা ভুল কিছু হবে না। টাইপস্ক্রিপ্টের ব্যবহার দিন দিন বৃদ্ধি পাবে। আর এটা শিখতে খুব বেশি সময়ও লাগবে না।

### টাইপস্ক্রিপ্টে যা যা শিখতে হবে

- Basic Types & Variable Declarations
- Classes
- Interfaces
- Generics
- Enums
- Advanced Types
- Iterators
- Generators
- Modules
- Namespaces
- Symbols
- Functions
- Type Inferences
- Decorators
- Mixins
- Triple Slash Directives
- Utility Types

[Visit Typescript Official Documentation](#)



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>

৭

---

অধ্যায় সাত

সাহিত্যের সৌন্দর্য

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ সিঙ্গেল পেজ অ্যাপ্লিকেশন
- ✓ SPA ফ্রেমওয়ার্কস
- ✓ ReactJS লাইব্রেরী
- ✓ ReactJS এ যা যা শিখতে হবে
- ✓ ReactJS টুলস

---

ডেভেলপমেন্টের জগতটা খুব বেশি পরিবর্তনশীল। আর এই ক্ষেত্রে ওয়েব ডেভেলপমেন্ট মনে হয় কয়েক ধাপ বেশি এগিয়ে আছে। কয়েক বছর আগের ওয়েব ডেভেলপমেন্ট টেকনিক আর এখনকার ওয়েব ডেভেলপমেন্ট টেকনিকের ভিতরেই আপনি আকাশ পাতাল ফারাক দেখতে পারবেন। এইতো কয়দিন আগেও মাল্টিপেজ ওয়েব অ্যাপ্লিকেশন ডেভেলপ করা হতো। প্রতিটা নেটওয়ার্ক রিকুয়েস্টের সাথেই একবার করে পেজ রিফ্রেশ করতে হতো সার্ভারের সাথে কমিউনিকেট করার জন্য। অ্যাপ্লিকেশনের সমস্ত লজিক এবং ডিজাইন পুরোটাই ম্যানেজ করতে হতো সার্ভার সাইডের অ্যাপ্লিকেশনের মাধ্যমে। ফ্রন্টেন্ড বলতে শুধু ছিল কিছু নির্জীব HTML, CSS ফাইলের টেম্পলেট। আমাদের সার্ভার বাবাজী প্রতিটা রিকুয়েস্ট পড়ে বুঝে সেই অনুযায়ী HTML ফাইল তৈরি করে ক্লায়েন্ট মানে ব্রাউজারের কাছে পাঠিয়ে দিত।

কিন্তু আজকে ওয়েব অ্যাপ্লিকেশন গুলো ঠিক এভাবে ডেভেলপড হয় না। ওয়েব অ্যাপ্লিকেশন ডেভেলপমেন্টের ধারাটা পুরোপুরি ভাবে পরিবর্তিত হয়ে গিয়েছে। এখন ওয়েব অ্যাপ্লিকেশন বলতে দুইটা ভিন্ন ভিন্ন অ্যাপ্লিকেশনকে বোঝায় - ফ্রন্টেন্ড এবং ব্যাকেন্ড। দুইটা অ্যাপ্লিকেশনই সম্পূর্ণ ভিন্ন ভিন্ন ভাবে ভিন্ন প্রোসেসের মধ্য দিয়ে তৈরি করতে হয়। ফ্রন্টেন্ড অ্যাপ্লিকেশন গুলো এখন স্ট্যান্ডএলোন অ্যাপ্লিকেশন, যার সাথে ব্যাকেন্ডের কোনো সম্পর্ক নেই। আপনি কোন টেকনোলজি, কোন ল্যাংগুয়েজ ব্যবহার করে আপনার সার্ভার অ্যাপ্লিকেশনটা ডেভেলপ করেছেন, কোথায় আপনার সার্ভার অ্যাপ্লিকেশনটাকে ডেপলয় বা হোস্ট করেছেন, কি লজিক লিখেছেন আপনার অ্যাপ্লিকেশনে, ফ্রন্টেন্ড অ্যাপ্লিকেশনের কোনো কিছুতেই যায় আসে না। সে শুধু আপনার কাছে এক্সপেন্ট করে তার প্রয়োজন অনুযায়ী ডাটা আপনি তাকে প্রোভাইড করবেন এবং যেই ডাটা গুলো সার্ভারে জমিয়ে রাখা দরকার সেই ডাটা গুলো সে আপনাকে কোনো না কোন ভাবে পাঠিয়ে দিবে আর আপনি সেগুলোকে স্টোর করে রাখবেন। এই পুরো কাজটা যেই সিস্টেমে ঘটবে তার নাম হচ্ছে REST API.

বর্তমানে ফ্রন্টেন্ড অ্যাপ্লিকেশনের জানার দরকার নেই এর ব্যাকেন্ড কে, আবার ব্যাকেন্ড অ্যাপ্লিকেশনেরও জানার দরকার নেই যে এর ফ্রন্টেন্ড কে বা কোথায় আছে। দুইটা অ্যাপ্লিকেশন সম্পূর্ণ ভিন্ন ভিন্ন সার্ভারে আবার ছোট অ্যাপ্লিকেশনের ক্ষেত্রে একই সার্ভারে ডেপলয় করা হয়। এই ধরনের ফ্রন্টেন্ড অ্যাপ্লিকেশন গুলোকে বলা হয় সিঙ্গেল পেজ অ্যাপ্লিকেশন। এই নামটা শুনে অনেকেই সিঙ্গেল পেজ পোর্টফলিও সাইটের মতো ওয়েবসাইটের কথা মনে করে থাকে। একসময় আমিও তাই মনে করতাম। একটা কথা আবার

---

স্মরণ করিয়ে দেই, আমি কিন্তু এখানে ওয়েবসাইট নিয়ে কোন কথা বলছি না, বলছি ওয়েব অ্যাপ্লিকেশন নিয়ে। ওয়েবসাইট আর ওয়েব অ্যাপ্লিকেশন কখনোই এক জিনিস না।

সিঙ্গেল পেজ ওয়েব অ্যাপ্লিকেশনেও আপনি অনেক গুলো পেজ নিয়ে কাজ করতে পারবেন। পেজ গুলোর ভিতরে লিংকিং করতে পারবেন, এক পেজ থেকে অন্য পেজে ভিসিট করতে পারবেন। প্রতিটা পেজের নিজস্ব লিংকও থাকবে আর আপনি সেই লিংক শেয়ারও করতে পারবেন। তাহলে এর নাম সিঙ্গেল পেজ অ্যাপ্লিকেশন কেন? কারণ যত বড় অ্যাপ্লিকেশনই হোক না কেন পুরো অ্যাপ্লিকেশনে একটি মাত্র HTML ফাইল থাকে। যেখানে মাল্টিপেজ অ্যাপ্লিকেশনে প্রতিটা পেজের জন্য একটা করে HTML ফাইলের দরকার হয়, সেখানে পুরো অ্যাপ্লিকেশনের সব গুলো ফাইলের জন্য একটি মাত্র HTML ফাইল? তাহলে বাকি পেজ গুলো কিভাবে রেন্ডার হবে?

বাকি পেজ গুলো রেন্ডার হবে না। এই বিষয়টা ভালোভাবে বুঝতে হলে আপনাকে বুঝতে হবে মাল্টিপেজ অ্যাপ্লিকেশন কিভাবে কাজ করে থাকে? একটা মাল্টিপেজ অ্যাপ্লিকেশনে যখন আমরা কোনো লিংকে ক্লিক করি তখন আমাদের ব্রাউজার ওই পেজ রেন্ডার করার জন্য প্রয়োজনীয় HTML কোড চেয়ে সার্ভারের কাছে রিকুয়েষ্ট করে। যখন সার্ভার রিকুয়েষ্ট ব্যাক করে অর্থাৎ রেসপন্স পাঠায় তখন ব্রাউজার ওই HTML কোডটাকে নতুন করে রেন্ডার করে আমাদেরকে দেখায়। এখানে প্রতিটা পেজের জন্য নতুন করে HTML, CSS, Javascript বা প্রয়োজনীয় সমস্ত ফাইলের জন্য ব্রাউজারকে রিকুয়েষ্ট করতে হয় এবং রেসপন্স আসলে নতুন করে রেন্ডার করতে হয়। কিন্তু সিঙ্গেল পেজ অ্যাপ্লিকেশনে ঠিক তার উল্টোটা ঘটে থাকে।

আমরা প্রথমবার যখন সার্ভারের কাছে রিকুয়েষ্ট পাঠাবো, সহজ কথায় বললে প্রথমবার যখন ওয়েব অ্যাপ্লিকেশনটাতে ভিসিট করবো তখন সার্ভার পুরো অ্যাপ্লিকেশনটাকেই একবারে ব্রাউজারের কাছে পাঠিয়ে দেবে। সেখানে একটা মাত্র HTML ফাইল থাকবে এবং সেটাও ব্ল্যাংক বা ফাকা ফাইল। ব্রাউজার তখন এই ব্ল্যাংক HTML ফাইলটাকেই রেন্ডার করবে যা সম্পূর্ণ ফাকা থাকবে। একবার ব্রাউজার HTML ফাইলটা রেন্ডার করতে পারলেই HTML ফাইলের কাজ শেষ, তখন পুরো কাজটা করবে জাভাস্ক্রিপ্ট। জাভাস্ক্রিপ্ট AJAX এর মাধ্যমে ব্যাকেন্ড সার্ভারের সাথে কমিউনিকেট করে ডাটা এর জন্য রিকুয়েষ্ট করবে। ডাটা যখন চলে আসবে তখন DOM ম্যানিপুলেশনের মাধ্যমে সদ্যপ্রাপ্ত ডাটা ব্রাউজারে শো করানো হবে। পুরো প্রোসেসটা এত দ্রুত ঘটে যাবে যে আপনি ব্ল্যাংক পেজ কখনো লক্ষ্যই করবেন না, তবে বেশিরভাগ ক্ষেত্রেই একটা আকর্ষণীয় লোডিং স্পিনার দিয়ে বুঝিয়ে দেওয়া হয় যে পেজ লোড

---

নিচে, একটু ধৈর্য ধরুন। এভাবেই সিঙ্গেল পেজ অ্যাপ্লিকেশন গুলো কাজ করে থাকে। যখন আপনি অন্য পেজে ভিসিট করার জন্য কোন একটা লিংকে ক্লিক করবেন তখনও লক্ষ্য করে দেখবেন ব্রাউজার সার্ভারের কাছে কোনো রকম কোনো রিকুয়েস্ট পাঠাচ্ছে না, পেজ রিফ্রেশ নিচে না। তাহলে নতুন পেজ কিভাবে রেন্ডার হচ্ছে? কোন লিংকে ক্লিক করলে কোন পেজটা ওপেন করতে হবে, সেখানে কি কি ডাটা থাকবে তাতে আগে থেকেই আপনার অ্যাপ্লিকেশনে বলা থাকবে। জাভাস্ক্রিপ্ট যখন কোনো লিংকে ক্লিক হতে দেখবে তখন আগের রেন্ডার হওয়ার DOM এলিমেন্ট গুলোকে মুছে দিয়ে নতুন পেজের DOM এলিমেন্ট গুলোকে রেন্ডার করবে এবং প্রয়োজনীয় নেটওয়ার্ক কল সম্পর্ক করে আপনার সামনে নতুন পেজটা ওপেন করবে।

সিঙ্গেল পেজ অ্যাপ্লিকেশনের পুরো গেমটাই জাভাস্ক্রিপ্টের হাতে। আপনি যদি মনে করেন জাভাস্ক্রিপ্ট ব্যবহার করে কোনো রকম কোনো ফ্রেমওয়ার্ক বা লাইব্রেরী ব্যবহার না করেই সিঙ্গেল পেজ অ্যাপ্লিকেশনের মত অ্যাপ্লিকেশন বানাবেন তাহলে তা সম্ভব। তবে তার জন্য আপনার জাভাস্ক্রিপ্ট এবং DOM সম্পর্কে বিস্তর একটা জ্ঞানের দরকার হবে। এবং অবশ্যই এটা সময় সাপেক্ষ একটা বিষয়। শেখার সময় আপনি এই রকম কাজ করে নিজের দক্ষতা যাচাই করতেই পারেন। কিন্তু বাস্তব জীবনের প্রোজেক্ট গুলোর জন্য ভালো হয় আপনি একটা নির্ভরযোগ্য ফ্রেমওয়ার্ক বা সিঙ্গেল পেজ অ্যাপ্লিকেশন বানানো যায় এই রকম একটা লাইব্রেরী নিয়ে কাজ করলেন। সিঙ্গেল পেজ অ্যাপ্লিকেশন ডেভেলপ করার জন্য আপনার হাতের কাছেই আপনি অনেকগুলো নির্ভরযোগ্য লাইব্রেরী এবং ফ্রেমওয়ার্ক পাবেন। কিন্তু আমি একজন বিগিনারকে সব সময়ের জন্যই সার্জেন্ট করবো ReactJS কে।

React আমার শেখা প্রথম ফ্রন্টেন্ড টুল না। আমি ভক্ত ছিলাম Angular এর। Angular একটা বড়সড় ফ্রন্টেন্ড ফ্রেমওয়ার্ক। আর আমি হচ্ছি একজন Solo ডেভেলপার। যার ফলে Angular অনেক মজার হলেও ম্যানেজ করতে অনেক কষ্ট হতো। React এর জনপ্রিয়তা দেখে একবার চেষ্টা করলাম এটা শেখার। কিন্তু এর আজব সিনট্যাক্স দেখে প্রথমেই ভড়কে গেলাম। এর পরে কিছু দিন Vue নিয়ে কাজ করলাম। Vue সিম্পল, সহজ এবং আমার জন্য অনেক সহজই ছিল। যেহেতু Angular থেকে এসেছি, সেহেতু সব কিছুই কেমন যেন পরিচিত লাগছিল। তবে Angular থেকে এসে Vue তে আমি খুব একটা মজা পাইনি। Angular এ যখন কাজ করেছি তখন Typescript নিয়ে কাজ করতাম। Angular কাজ করে MVC (Model View Controller) আর্কিটেকচার মডেলে যার সাথে আমি পূর্ব পরিচিত এবং কমফোর্টেবল। Angular এর আর একটা প্লাস পয়েন্ট হচ্ছে Dependency

---

Injection এবং Observable Pattern. একজন জাভা প্রোগ্রামার হিসেবে Angular কে ভালোবাসার জন্য এই কারণ গুলোই আমার কাছে যথেষ্ট ছিল। Angular কে আমার অনেক গুছানো মনে হয়েছে এবং যেহেতু এটা একটা ফ্রেমওয়ার্ক তাই একটা ফ্রন্টেন্ড অ্যাপ্লিকেশন ডেভেলপ করার জন্য যা যা দরকার তার সব কিছু আপনি এর সাথে পাবেন। এর সাথে সাথে Angular এর একটা পাওয়ারফুল CLI বা কমান্ড লাইন টুলসও ছিল যা তখনও Vue এর ছিল না। Angular এর সাথে কম্পেয়ার করে তখন Vue কে আমার খুব একটা পছন্দ হয় নি। কিন্তু Solo ডেভেলপার হিসেবে একে ম্যানেজ করাও আমার জন্য কষ্টসাধ্য হয়ে যাচ্ছিল। আমার একটা লাইট ওয়েট সমাধানের দরকার ছিল। তাই এক রকম বাধ্য হয়েই আবার React শেখার চেষ্টা করলাম।

এবার আমি খুব মজা পেলাম, যেই আমি React এর অন্তর্ভুক্ত সিনট্যাক্স দেখে বিশেষ করে JSX দেখে ভড়কে গিয়ে React শিখতে চাছিলাম না, সেই আমিই শুধুমাত্র JSX এর প্রেমে পড়ে সিদ্ধান্ত নিলাম যে React আমি করবোই। কিছু দিন সময় দিয়ে React কে ভালো ভাবে বোঝার চেষ্টা করলাম। React যেহেতু একটা ছোট্ট লাইব্রেরী তাই এখানে Angular এর মত বড় বড় সব ডিজাইন প্যাটার্ন নেই, বিউল্টইন ভাবে প্রতিটা সন্তান সমস্যার জন্য সমাধান দেওয়া নেই, দেওয়া নেই কোনো ইউটিলিটি টুলস। তারপরেও একটা লাইব্রেরী এত সুন্দর কিভাবে হতে পারে, কিভাবে এত সহজে ইন্টারেক্টিভ ইন্টারফেস ডিজাইন করতে পারে তা কোনো ভাবেই আমার বুঝে আসছিল না। React, Angular এর মত এত অর্গেনাইজড না, কিন্তু খুব বেশি ফ্লেক্সিবল, সর্বোচ্চ স্বাধীনতা দেওয়া আছে আমার হাতে। React এর মত Vue ও অনেক বেশি ফ্লেক্সিবল কিন্তু আমি Vue এর ভিতরে তেমন কোন নতুনত্ব দেখতে পাইনি। নতুনত্বের ছোঁয়া আমি পেয়েছি React করতে এসে। আমার মনে হয়েছে কেন আমি এত দিন React এর থেকে দূরে ছিলাম। কর্ম জীবনে আপনাকে সব কিছু নিয়েই কাজ করতে হবে, আমরাও প্রয়োজনের তাগিদে সব গুলো ফ্রেমওয়ার্ক নিয়েই কাজ করে থাকি। কিন্তু ভালো লাগার জায়গা বলেও একটা বিষয় থাকে, যেই জায়গাটা React দখল করে নিয়েছে। আমার কাছে মনে হয়েছে বিগিনারদের জন্য সব থেকে সহজ হচ্ছে React, নতুন করে খুব বেশি কিছু শেখার দরকারই নেই। যদি আপনি জাভাস্ক্রিপ্ট জানেন তাহলে React এ কাজ শিখতে খুবই অল্প সময় লাগবে।

React হচ্ছে জাভাস্ক্রিপ্টের একটা লাইব্রেরী যা ব্যবহার করে আপনি খুব সহজে ইন্টারেক্টিভ ইউজার ইন্টারফেস তৈরি করতে পারেন। React সম্পর্কে বলার মত আর কোন কথা নেই। React কে আর কোনো ভাবে ডিফাইন করা সম্ভব না। বড় বড় অ্যাপ্লিকেশন

---

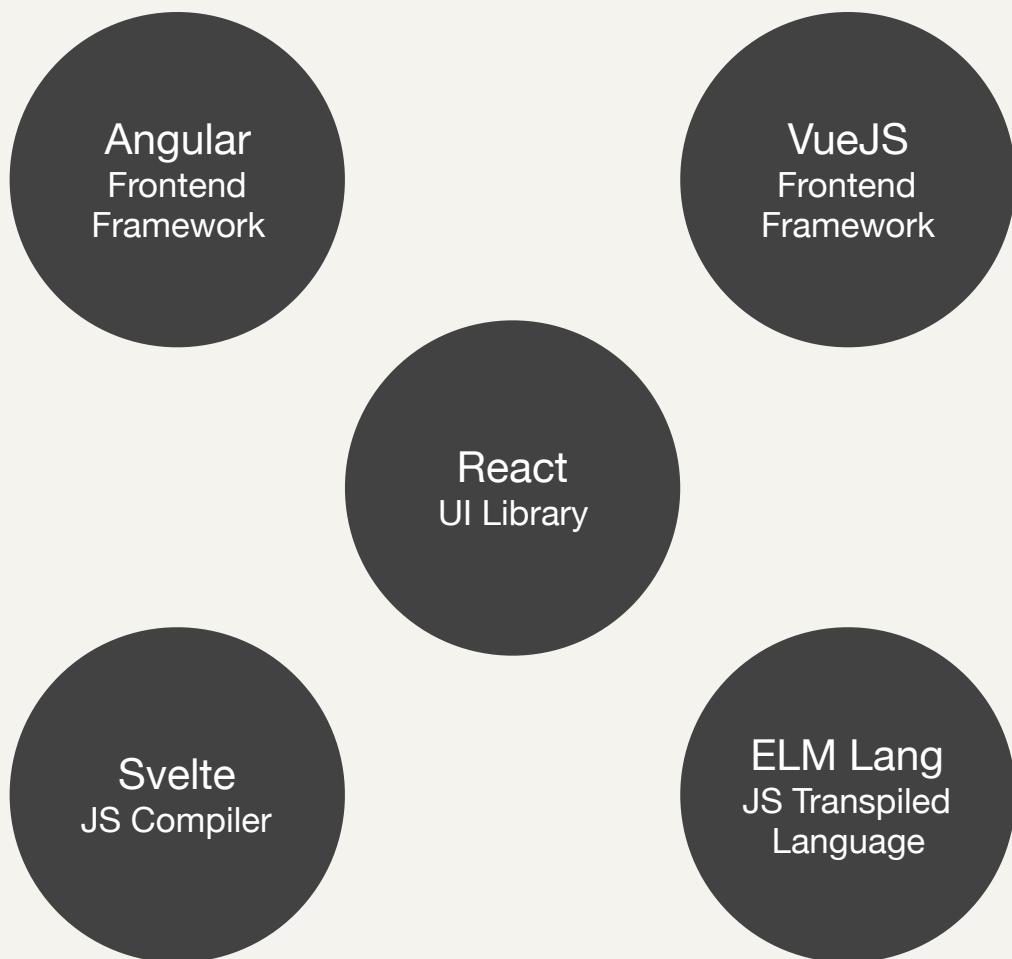
ডেভেলপ করার জন্য React এর জন্ম না, এর জন্ম একটা বড় অ্যাপ্লিকেশনের ছোট্ট একটা অংশকে ইন্টারেক্টিভ করার জন্য। পূর্বে JQuery ব্যবহার করে যেই কাজ করতাম সেই একই কাজ নতুন ভাবে এফিসিয়েন্ট ওয়েভেতে করার কাজটা করছে React. তাই একটা মাল্টিপেজ অ্যাপ্লিকেশনও আপনি সহজেই React ব্যবহার করতে পারবেন। React যেহেতু একটা লাইব্রেরী, তাই এর লার্নিং কার্ড খুব ছোট। React এর মুখ্য বিষয় হচ্ছে কম্পানেন্ট, এটি কাজ করে ছোট ছোট কম্পানেন্ট তৈরির করার মাধ্যমে। প্রতিটা কম্পানেন্টের নিজস্ব ডাটা আছে যার নাম স্টেট, এই ডাটা ম্যানিপুলেট করারও নিজস্ব ওয়ে আছে। একটা ইন্টারফেস তৈরির ক্ষেত্রে এরকম অসংখ্য কম্পানেন্ট তৈরি করতে হয় আর সেগুলোকে কম্পোজ করার মাধ্যমে একটা বড় ইন্টারফেস তৈরি করা হয়। একটা কম্পানেন্ট থেকে আর একটা কম্পানেন্টে ডাটা পাস করার জন্য ব্যবহৃত হয় প্রপস। আর ডাটা গুলো কন্ডিশনালই রেন্ডার করার জন্য আছে JSX. এর সাথে কম্পানেন্টের কিছু ফাংশনালিটিস আছে। ব্যস React শেষ, React এ এর থেকে বেশি কিছুই নেই।

React একটা লাইব্রেরী, তাই শুধুমাত্র React ব্যবহার করে আপনি সিঙ্গেল পেজ অ্যাপ্লিকেশন বানাতে পারবেন না। এর জন্য আপনার দরকার পড়বে অসংখ্য লাইব্রেরী। আমরা অনেকেই মনে করি, React শেখার অর্থ হচ্ছে ফ্রন্টেন্ড ডেভেলপমেন্ট শেখা। কিন্তু ব্যাপারটা সম্পূর্ণ ভুল। React একটা ছোট্ট লাইব্রেরী যা আপনাকে ইউজার ইন্টারফেস তৈরিতে সাহায্য করবে। আর ফ্রন্টেন্ড ডেভেলপমেন্ট হচ্ছে একটা প্রোসেস। এই প্রোসেস সম্পূর্ণ করতে আপনার যেমন অসংখ্য থার্ড পার্টি টুলস এর প্রয়োজন হবে, তেমনি আপনার প্রয়োজন হবে অসংখ্য ডেভেলপমেন্ট থিওরির। আপনার জানার দরকার হবে ডেভেলপমেন্ট বেষ্ট প্রাক্টিসেস, দরকার হবে ডিজাইন প্যাটার্নস। যার বেশির ভাগই অর্জন করতে হয় অভিজ্ঞতা থেকে।

একটা ফ্রন্টেন্ড অ্যাপ্লিকেশনের সব থেকে গুরুতপূর্ণ তিনটা কাজ হচ্ছে স্টেট ম্যানেজ করা, ক্লাইন্ট সাইড রাউটিং হ্যান্ডেল করা এবং AJAX এর মাধ্যমে সার্ভারের সাথে কমিউনিকেশন সম্পূর্ণ করা। সাধারণত এই কাজ গুলো সম্পূর্ণ করার জন্যই আমরা থার্ড পার্টি সল্যুশনের কাছে যাই। তবে আপনি চাইলে থার্ড পার্টি সল্যুশনের কাছে না গিয়ে নিজে কোড করেও এর সমাধান করতে পারেন, তবে এটা হবে খুবই জঘন্য একটা আইডিয়া। এর থেকে ভাল হয় থার্ড পার্টি বহুল ব্যবহৃত ইউটিলিটি লাইব্রেরী গুলো ব্যবহার করলে।

আপনি ফ্রন্টেন্ড ডেভেলপমেন্ট শিখছেন মানে অ্যাপ্লিকেশন ডেভেলপমেন্ট শিখছেন। হয়তু কয়েকদিন টুলস গুলো নিয়ে একটু ঘাটাঘাটি করলেই আপনি ছোটোখাটো কিছু অ্যাপ্লিকেশন ডেভেলপও করতে পারবেন। তবে একজন ভালো মাপের ডেভেলপার হওয়ার জন্য আপনাকে সময় ব্যয় করতে হবে। ভিন্ন ভিন্ন ধরনের অ্যাপ্লিকেশন ডেভেলপ করার মাধ্যমে নতুন নতুন এক্সপেরিয়েন্স অর্জন করতে হবে। পূর্বে ফ্রন্টেন্ড লজিক অ্যাপ্লাই করার খুব বেশি স্কোপ ছিল না। কিন্তু এখন বিষয়টা পুরোপুরি বিপরীত। একটা ডেস্কটপ অ্যাপ্লিকেশন, মোবাইল অ্যাপ্লিকেশন তৈরি করতে আপনার যতটা ডেডিকেশন দরকার হয় ঠিক ততটা ডেডিকেশনই আপনার দরকার হবে। ঠিক ততটাই লজিক ডেভেলপ করার দরকার হবে ফ্রন্টেন্ড ডেভেলপমেন্টের ক্ষেত্রে। কারণ বর্তমানে এগুলো সবই প্রায় একই রকম এবং একই ওয়েভেতেই কাজ করে থাকে।

## কিছু ফ্রন্টেন্ড লাইব্রেরী, ফ্রেমওয়ার্ক এবং কম্পাইলারস



## ReactJS এর কোর ফিচারস যা যা শিখতে হবে

- Component Theory
- Component Tree & Props
- Component State and API
- JSX (Javascript Extension)
- Conditional Rendering
- Rendering Lists & Tables
- Event Handling
- Input Elements
- Two Way Data Binding
- React Forms
- Component Lifecycle Methods
- Component Design Patterns
- Context API
- React Hooks
- Refs and Forward Refs
- Error Boundaries
- Performance Optimisation Techniques
- Test React Applications

## অন্য যেই টুলস গুলো লাগবে সম্পূর্ণ অ্যাপ ডেভেলপমেন্ট করতে

- Material UI (Component Library)
- Ant Design (Component Library)
- Reactstrap (Component Library)
- React Router (Routing)
- Reach Router (Routing)
- Axios (AJAX)
- Redux (State Management)
- React Redux (Redux Bindings)
- MobX (State Management)
- Easy Peasy (State Management)
- Redux Form
- Formik
- React Hook Form
- Use React (use-react)
- React DND (Drag and Drop)
- React Spring (Animation)
- React Transition Group (Animation)
- React Developer Tool (Extension)
- Redux Developer Tool (Extension)

# React Redux শিখার রেফারেন্স

## Must Read React Redux Books:

- Learning React: Fundamental Web Development with React Redux By Alex Banks
- React: Up & Running: Building Web Applications By Stoyan Stefanov
- Building ReactJS Applications with Redux By David Geary
- Redux in Action By Garreau and Will Faurot
- Learn React Hooks: Build and Refactor Modern ReactJS Applications Using Hooks By Daniel Bugl
- Mastering React Test Driven Development By Daniel Irvine

## Youtube Channels to React Redux

- [Digitaloy](#) (Bangla)
- [Learn Hunter](#) (Bangla)
- [Stack Learner](#) (Bangla)
- [Codevolution](#) (English)
- [Free Code Camp](#) (English)
- [Learn Code Academy](#) (English)

## Websites to Learn React Redux

- [React Official Docs](#)
- [Redux Official Docs](#)
- [ReactJS Examples](#)
- [React Design Patterns](#)
- [React For Beginners - Wes Bos](#)

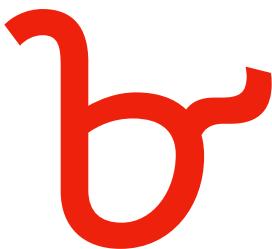
## Training Program for React Redux

- [Stack Learner Premium Courses](#)
- [Stack Learner Offline Bootcamps](#)



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>



---

অধ্যায় আট

## সাহিত্যের মেরুদণ্ড

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ ওয়েব ডেভেলপমেন্টের ভূল ধারণা
- ✓ ব্যাকেন্ড ডেভেলপমেন্ট
- ✓ বেস্ট ব্যাকেন্ড ল্যাংগুয়েজ
- ✓ ব্যাকেন্ড ল্যাঙ্গুয়েজেস এবং ফ্রেমওয়ার্কস

---

কেউ যখন প্রথম HTML শিখা শুরু করে তখনই তার মনের মধ্যে একটা স্বপ্নের সংগ্রাম হয়। একদিন আমি একজন বড় ডেভেলপার হব। অনেক বড় বড় সব ওয়েব অ্যাপ্লিকেশন ডেভেলপ করবো। লক্ষ লক্ষ মানুষ আমার তৈরি করা অ্যাপ্লিকেশন ব্যবহার করবে। আমি যখন প্রোগ্রামিং ল্যাংগুয়েজ শিখা শুরুই করিনি তখন থেকেই আমার স্বপ্ন আমি ওয়েব ডেভেলপার হব। তখন অবশ্য ওয়েব ডেভেলপমেন্ট বলতে ব্যাকেন্ড ডেভেলপমেন্ট বা সার্ভার সাইড অ্যাপ্লিকেশনকেই বোঝাতো। কারণ তখনও ফ্রন্টেন্ড ডেভেলপমেন্ট টার্মসটার সাথে সবাই পরিচিত ছিলাম না। ডেভেলপমেন্ট বলতে শুধু ব্যাকেন্ড ডেভেলপমেন্টকেই আমরা বুঝতাম। কিন্তু মজার বিষয় হচ্ছে ৯-১০ বছর পরে এসেও আমাদের দেশে বেশিরভাগ মানুষ ডেভেলপমেন্ট বলতে ব্যাকেন্ড ডেভেলপমেন্টকেই বোঝে, আর ওয়েব ডিজাইনকেই ফ্রন্টেন্ড ডেভেলপমেন্ট মনে করে। খুব সীমিত সংখ্যক মানুষ ব্যতীত বেশিরভাগ মানুষই তাদের স্কিল আপগ্রেড করতে পারিনি এখনো, এটা খুবই হতাশাজনক।

আমাদের দেশে ওয়েব ডেভেলপমেন্ট কথাটার সাথে আরও দুইটা নাম অন্তরঙ্গ ভাবে জড়িত, আর তা হচ্ছে পিএইচপি এবং ওয়ার্ডপ্রেস। কেউ ওয়েব ডেভেলপার হতে চাচ্ছে মানে তাকে এই পিএইচপি এবং ওয়ার্ডপ্রেসই শিখতে হবে এইরকম একটা গুজব বহু বছর আগে থেকেই শুরু হয়েছিল, কিন্তু দুঃখের বিষয় গুজবটা এতটাই পাওয়ারফুল ছিল যে আজ পর্যন্ত মানুষ এই গুজোব মনেই ডেভেলপমেন্ট শিখছে। একটা সময় আমাদের দেশে পিএইচপি এবং ওয়ার্ডপ্রেস রাজস্ব করেছে, এখনো করছে, ভবিষ্যতেও করবে। একজন ডেভেলপারের পিএইচপি ভালো লাগতেই পারে, ওয়ার্ডপ্রেস ব্যবহার করে যদি সে সহজেই একটা পার্সোনাল ওয়েব সাইট তৈরি করতে পারে, Woocommerce ব্যবহার করে যদি সে একটা ছোটখাটো কোম্পানির অল্প কিছু ইউজারের জন্য একটা ইকমার্স সাইট তৈরি করে দিতে পারে তাহলে ক্ষতি কি? কোনো ক্ষতি নেই। কাজ করাতে কোনো ক্ষতি নেই, ক্ষতি হচ্ছে এটা বিশ্বাস করাতে যে পিএইচপি ব্যতীত কোনোভাবেই ওয়েব ডেভেলপমেন্ট করা যায় না।

আমরা সবাই জানি ফ্রিলান্সিং সাইট গুলোতে পিএইচপি এবং ওয়ার্ডপ্রেস এর কাজ বেশি পাওয়া যায়। আমাদের দেশে যখন ফ্রিলান্সিং ব্যাপারটা যাত্রা শুরু করল, তখন অনেক ডেভেলপার এই পিএইচপি এবং ওয়ার্ডপ্রেসের কাজ করে অনেক সফলতা অর্জন করল, দেশের মুখ উজ্জ্বল করলো। প্রচুর ছোট ছোট আইটি ফার্ম তৈরি হল যারা ফ্রিলান্সিং করে দেশে বৈদেশিক মুদ্রা আনতে থাকলো। যেহেতু ফ্রিলান্সিং সাইটগুলোতে পিএইচপি এর কাজ বেশি পাওয়া যায় তাই তারা প্রচুর পিএইচপি ডেভেলপার তৈরি করা শুরু করল। এর মাঝেখানেই কখন মানুষের মনের ভিতরে কোথায় যেন গেঁথে গেল ওয়েব ডেভেলপমেন্ট মানেই পিএইচপি।

---

এই হাইপের কারণে অভিজ্ঞদের কোনো প্রশ্নে হল না, কারণ তারা একটা শক্ত অবস্থানেই রয়েছেন। নতুনরা পরে গেল বিপদে। চাহিদার তুলনায় কয়েকগুণ বেশি পিএইচপি ডেভেলপার তৈরি হয়ে গেল। যাদের সবার কর্মসংস্থান করা ফ্রিলান্সিং সাইট গুলোর পক্ষেও সম্ভব না, আমাদের দেশের আইটি ফার্ম গুলোর পক্ষেও সম্ভব না। কারণ আইটি ফার্ম গুলোতে শুধু পিএইচপি নিয়ে কাজ করে না, তাদের ভিন্ন ভিন্ন টেকনোলজি নিয়ে কাজ করতে হয়। দেশে পিএইচপি ডেভেলপারের কোনো অভাব নেই, প্রতিনিয়ত নতুন নতুন পিএইচপি ডেভেলপার হয়েই যাচ্ছে। কিন্তু জাভা, পাইথন, নোডজেএস, সি শার্প ডেভেলপার খুঁজতে গেলে হাতে গোনা কয়েকজনকেই পাওয়া যায়। যদি সবাই পিএইচপি নিয়ে কাজ করতে চাই, তাহলে এত কাজ কিভাবে তৈরি হবে? বরঞ্চ যারা ভালো পিএইচপি ডেভেলপার আছেন, ভালো ভালো কাজ করেন তারা বিপদে পড়ে যাচ্ছেন। কারণ নতুনরা বাজারে এসেই দ্রুত কাজ পাওয়ার উদ্দেশ্যে কাজের মূল্য কমিয়ে ফেলছে। সাথে সাথে গুণগত মানও কমিয়ে ফেলছে।

ওয়েব ডেভেলপমেন্ট মানেই পিএইচপি, এই একটা হাইপ কি পরিমাণ ক্ষতি করেছে একটু হিসেব করলেই আপনি বুঝতে পারবেন। পিএইচপি ডেভেলপারের সংখ্যা এত আমাদের দেশে যে ১০ হাজার টাকা স্যালারিতেও মানুষ ডেভেলপমেন্টের কাজ করতে রাজি হয়ে যায়। যারা অভিজ্ঞ, অনেক দিন থেকে মার্কেট প্লেসে কাজ করছে, বা জব করছে তাদের কাজের কোনো সমস্যা হবে না। কিন্তু যে কিছু দিন হলো পিএইচপি শিখেছে, বা আজকে পিএইচপি শেখা শুরু করছে একটা সিকিউরড লাইফের আশায়, কিভাবে তারা আপনাদের মত অভিজ্ঞ ব্যক্তিদেরকে টপকে নিজের একটা অবস্থান করে নেবে? যেখানে অভিজ্ঞ ব্যক্তিরায় অনেক ক্ষেত্রে ঘরে বসে আছেন, নতুন নতুন টেকনোলজি শিখে নিজেকে আপগ্রেড করার চেষ্টা করছেন। তর্কের খাতিরে আমি ধরে নিলাম ১০% নতুন ডেভেলপার নিজের জায়গা ঠিকই করে নেবে, কিন্তু বাকি ৯০% এর কি হবে? তারা এত কষ্ট করে ডেভেলপমেন্টে নিজের বেস তৈরি করলো কি বেকার বসে থাকার আশায়? এই অবস্থায় কি তারা নতুন করে কোনো আপডেটেড টেকনোলজি শেখার মানুষিকতাও রাখে? এই মানুষ গুলোই যদি এই সময় গুলো অন্য টেকনোলজি গুলো শিখত যেখানে কম্পিউটিশন অনেক কম কিন্তু ডিমান্ড অনেক বেশি তাহলে কি তার সময়ের সঠিক ব্যবহার করা হতো না? তাহলে কি দেশে আরও কিছু বেশি বৈদেশিক মুদ্রা আসার ক্ষেত্র তৈরি হতো না? আমাদের দেশেও কিছু স্বপ্নবাজ মানুষ কি বড় বড় সফটওয়্যার কোম্পানী দিয়ে নিজের সফটওয়্যার সার্ভিস বিক্রি করার স্বপ্ন দেখত না? ইন্ডিভিজুয়াল ফ্রিলান্সার তৈরি না হয়ে হাজার হাজার এজেন্সি তৈরি হতো না? সফটওয়্যার খাত থেকে আমাদের দেশ কি সব থেকে বেশি বৈদেশিক মুদ্রা অর্জন করতে পারতো না? অনেক কিছুই ঘটতে পারত, কিন্তু যার কোনটাই ঘটেনি। পূর্বে কি ঘটেছে সেটা যদি আমরা

---

ভুলেও যায়, বর্তমানকে দেখে চমকে উঠছি। কারণ বর্তমানেও মানুষ একই দিকেই ঝুঁকছে, এদের থামানোর কেউ নেই।

কয়েক বছর আগে যদি আমরা ভুল করতাম তাহলে বিষয়টা মনে নেওয়া যেত। কিন্তু আজকের দিনে দাঁড়িয়ে, যেখানে সবার হাতে হাতে স্মার্ট ফোন, সবার হাতে হাতে ইন্টারনেট, সবাই ইন্টারনেট ব্যবহার করতে জানি, এখন ভুল করলে সেটা মনে নেওয়া যায় না। সারাজীবন আমরা জেনে এসেছি, প্রথম বিশ্ব থেকে আমরা কয়েক বছর পিছিয়ে। অ্যাপল তার ফোন কবে রিলিজ দিবে, সেই ফোনে কি কি ফিচার থাকবে, এবাব আইফোনের ক্যামেরা কেমন, নতুন কি কি ফিচার যুক্ত হয়েছে প্রথম বিশ্ব থেকে বসে টিম কুক প্রেসেন্টেশন দিচ্ছে আর কয়েক মুভর্তের ব্যবধানে ঘরে বসেই আমরা সবাই তার লাইভ দেখতে পারছি। দুনিয়ার সমস্ত আপডেটেড ফোন, ল্যাপটপ, ক্যামেরা, গেম সব কিছুর আপডেট ইনস্ট্যান্টলি পেয়ে যাচ্ছি। পাবজি নতুন সেশন কবে দেবে তার জন্য দিন গুনতে পারছি, আপডেটের পরে খুঁজে খুঁজে তার প্যাচ নোট পড়তে পারছি। আর সব কিছু জানতে পারছি রিয়েল টাইমে, যখন প্রথম বিশ্বের মানুষ এই বিষয় গুলো সম্পর্কে জানছে ঠিক তখনই আমরা দ্বিতীয় তৃতীয় বিশ্বের মানুষ হয়েও একই সাথে জানতে পারছি।

অন্য সব সেক্টরের কথা বাদ দিলাম, কিন্তু কম্পিউটারের জগতে যদি আমরা পিছিয়ে থাকি, তাহলে আমি বলব আমরা পিছিয়ে থাকতে চাই তাই পিছিয়ে আছি। গুগলে একটু সার্চ করলেই আমরা জানতে পারি বর্তমান বিশ্ব কোন টেকনোলজি নিয়ে কাজ করছে, কোন টেকনোলজির ডিমান্ড বৃদ্ধি পাচ্ছে, কোন টেকনোলজিতে কত মানুষ কাজ করছে, কোন টেকনোলজিতে জবের থেকে কাজ করার মানুষের সংখ্যা বেশি আর কোথায় কাজ করার মানুষ নেই, কিন্তু টেকনজির অনেক ডিমান্ড। এলাকার সিএসই থেকে পাশ করে বের হওয়া বেকার বড় ভাইয়ের কাছ থেকেই কেন জানতে হবে যে আমার কি শেখা উচিত? না হয় জানলাম, দেশের সব থেকে বড় টেকনোলজিবিদের কাছ থেকেই জানলাম কোন বিষয়ে আমার ক্যারিয়ার গড়া উচিত, কিন্তু বাসায় এসে কি একবার ভ্যারিফাইও করে দেখব না যে কথাটা কতটা সঠিক?

এখন যদি আমার সাথে কোন ভুল হয়ে যায়, আজকে যদি কাজ শেখার পরেও আমাকে বেকার বসে থাকতে হয় সেই দোষ আমি কাউকে দিতে পারবো না। আমার ভুলের দ্বায়ভার সরকার নেবে না, আমার ভুলের দ্বায়ভার আমার ইউনিভার্সিটি নেবে না, কোনো ট্রেইনিং ইনসিটিউট বা কোনো ইউটিউব চ্যানেলের ওপরে আমি দোষ চাপাতে পারবো না। আমাকে

---

মনে নিতে হবে সেই দোষ পুরোটাই আমার, কারণ আমি আলসেমি করে একটু ঘেঁটে দেখিনি  
বিশ্ব কোন দিকে আগাছে আর আমি তার সাথে সাথে আগাছি কিনা?

ব্যাকেন্ড ডেভেলপমেন্ট হচ্ছে সব থেকে পাওয়ারফুল, সব থেকে ফ্লেক্সিবল এবং সব থেকে  
কনফিউসিং একটা ডেভেলপমেন্ট সেক্টর। এখানে আপনার সামনে অসংখ্য অপশন দেওয়া  
আছে, আপনার পছন্দ মত এবং প্রয়োজন মত টুলস আপনাকে বেছে নিতে হবে কোনো একটা  
কাজ সম্পন্ন করার জন্য। ফ্রন্টেন্ড ডেভেলপমেন্টের ক্ষেত্রে আমরা দেখছি আমাদের ওপরে  
জাভাস্ক্রিপ্টকে চাপিয়ে দেওয়া হয়েছে। কিন্তু ব্যাকেন্ড ডেভেলপমেন্টের ক্ষেত্রে সিদ্ধান্ত  
আপনাকেই নিতে হবে। আর এখানেই আমরা সব থেকে বড় ভুল কাজটা করে থাকি।

যে কোনো প্রোগ্রামিং ল্যাংগুয়েজ যা আপনি আপনার কম্পিউটারে রান করতে পারেন সেই  
ল্যাংগুয়েজ ব্যবহার করেই আপনি ব্যাকেন্ড ডেভেলপমেন্ট করতে পারবেন। অ্যাসেন্টলি, সি,  
সি++, সিশার্প, জাভা, পাইথন, রুবি, পার্ল, পি.এইচ.পি, জাভাস্ক্রিপ্ট যত প্রোগ্রামিং  
ল্যাংগুয়েজের নাম আপনি শুনেছেন, সব গুলো ল্যাংগুয়েজ ব্যবহার করেই আপনি সার্ভার  
কন্ট্রোল করার জন্য অ্যাপলিকেশন তৈরি করতে পারবেন, মানে ব্যাকেন্ডের কাজ করতে  
পারবেন। কেন পারবেন, সেটা বোঝার জন্য আপনাকে জানতে হবে সার্ভার কি, এবং কিভাবে  
কাজ করে?

সহজ কথায় সার্ভার হল একটা কম্পিউটার। এই কম্পিউটারটা পৃথিবীর যেকোন জায়গায়  
অবস্থান করতে পারে, এমনকি আপনার আমার বাসার ভিতরেও থাকতে পারে। এই  
কম্পিউটারটা ইন্টারনেটের সাথে কানেক্টেড থেকে, ২৪/৭ ওপেন থেকে আমাদের  
অ্যাপলিকেশন গুলোকে সার্ভ করছে। আর বাইরে থেকে আসা রিকুয়েস্ট গুলোকে হ্যান্ডেল  
করার জন্য এই কম্পিউটার গুলোর ভিতরে ইন্সটল করতে হয় বিভিন্ন সার্ভার সফটওয়্যার  
যেমন Apache, Nginx, Tomcat বা NodeJS. এই সার্ভার সফটওয়্যার গুলো রিকুয়েস্ট  
গুলোকে হ্যান্ডেল করে ২৪/৭ রানিং থাকা আমাদের অ্যাপলিকেশনের সাথে কানেক্ট করে দেয়।  
তখন আমাদের অ্যাপলিকেশন গুলো রিকুয়েস্টটা রিড করে এবং রিকুয়েস্টে যা করতে বলা  
হয়েছে সেই অনুযায়ী কাজ করে একটা রেসপন্স তৈরি করে। রেস্পন্স তৈরি হয়ে গেলে সার্ভার  
অ্যাপলিকেশন গুলো এই রেসপন্স আবার ইউজারের কাছে ব্যাক পার্টিয়ে দেয়। এই পুরো  
প্রোসেসটা ঘটে থাকে HTTP প্রোটোকলের মাধ্যমে।

---

আমাদের অনেকেরই ধারণা সার্ভার নিশ্চয় বডসড কোনো মেশিন যা স্যাটেলাইট এর সাথে কানেক্টেড থাকে। আসলে সার্ভার একটা সাধারণ কম্পিউটার। যেই কম্পিউটার আপনি প্রতিদিনের কাজে ব্যবহার করছেন সেই কম্পিউটারকেও সার্ভার হিসেবে ব্যবহার করা যাবে। কিন্তু যেহেতু ২৪/৭ এই কম্পিউটারকে ওপেন থাকতে হয়, তাই সার্ভার মেশিন গুলোর কম্পানেন্ট গুলো সাধারণত দামী এবং মজবুত দেখে নেওয়া হয়। সার্ভার কম্পিউটার অপারেট করার জন্যও কিন্তু অপারেটিং সিস্টেম এর দরকার পরে। আমরা যেই অপারেটিং সিস্টেম ব্যবহার করে প্রতি দিন কাজ করি, সেই একই অপারেটিং সিস্টেম সার্ভারেও ব্যবহার করতে পারি। তবে সার্ভারে আমরা যেহেতু অ্যাপিলিকেশন সার্ভ করা ব্যক্তিত অন্য কোনো কাজ করিনা, তাই সার্ভার অপারেটিং সিস্টেম গুলো এই কাজটা ভালো ভাবে করার জন্য আরও অপ্টিমাইজড করে তৈরি করা হয়। বাজারে আপনি যত গুলো লিনাক্স এর ডিস্ট্রিবিউশন দেখবেন সব গুলোরই প্রায় দুইটা ভার্শন দেখতে পারবেন। একটা ডেস্কটপ ভার্শন, অন্যটা সার্ভার ভার্শন। এমনকি উইন্ডোজেরও সার্ভার অপারেটিং সিস্টেম রয়েছে, কিন্তু সেটা লিনাক্সের মত ফ্রি না।

এখন নিশ্চয় বোঝা যাচ্ছে যে কেন, যে কোনো ল্যাংগুয়েজ ব্যবহার করেই আমরা সার্ভার সাইড অ্যাপলিকেশন বানাতে পারি? যেই ল্যাঙ্গুয়েজটা আমরা কম্পিউটারে রান করাতে পারি সেই ল্যাংগুয়েজটা আমরা সার্ভারেও রান করাতে পারব। সার্ভার আর কম্পিউটারের ভিতরে ব্যাসিক কোনো পার্থক্য নেই। তবে কিছু কিছু ল্যাংগুয়েজ তৈরিই করা হয়েছে ওয়েবে সার্ভার নিয়ে কাজ করার জন্য, যেমন পি.এইচ.পি। পি.এইচ.পি এর পূর্ণরূপ প্রথমে ছিল পার্সনাল হোম পেজ। যখন প্রথম ওয়েবের আবির্ভাব হল তখনই পার্সনাল হোম পেজ তৈরি করার জন্য পি.এইচ.পি এর আবিষ্কার করা হয়। পার্সনাল ওয়েবসাইট, বিজনেস ওয়েবসাইট বা এমন ওয়েব সাইট যা শুধু ইনফরমেশন শেয়ার করার কাজে ব্যবহৃত হবে এমন সকল ওয়েব সাইট বানানোর জন্য পি.এইচ.পি এর বিকল্প আজও নেই। পি.এইচ.পি ব্যবহার করে খুব স্বল্প জ্ঞান কাজে লাগিয়ে, খুব স্বল্প সময়ে আপনি সকল ধরনের ওয়েবসাইট বানাতে পারবেন।

২০০৩ সালে ওয়েবডেভেলপমেন্টের ক্ষেত্রে পি.এইচ.পি এর একটা নতুন মাত্রা যুক্ত হয়। কারণ এবছরেই বাজারে আসে দুনিয়ার সব থেকে জনপ্রিয় সিএমএস ওয়ার্ডপ্রেস। ওয়েবের যাত্রার শুরুর দিকে কেউ কল্পনায় করেনি যে, ওয়েবকে ব্যবহার করে আজকের দুনিয়ার প্রায় সব ধরনের সফটওয়্যার সার্ভিসই প্রদান করা সম্ভব হবে। তখন ওয়েব ছিল ইনফরমেশন শেয়ার করার একটা মাধ্যম মাত্র। পার্সনাল ব্লগসাইট, নিউজ সাইট, সরকারি সাইটের মত সব সাইট তখন তৈরি হতো। ওয়ার্ডপ্রেস যখন আসে তখন ফেসবুকও আবিষ্কার হয় নি।

---

পার্সোনাল সাইটগুলো পিএইচপি ব্যবহার করে খুব সহজে ডেভেলপতো করা যায়, কিন্তু এর কন্টেন্ট ম্যানেজ করার জন্য অন্য আর একটা অ্যাপলিকেশনের দরকার পরে। এই সমস্যাটা বুঝতে পেরে ওয়ার্ডপ্রেস একটা কন্টেন্ট ম্যানেজমেন্ট সিস্টেম বা CMS তৈরি করে যা আজ পর্যন্ত জনপ্রিয়। বিশ্বে যত ওয়েবসাইট আছে তার বেশির ভাগ ওয়েবসাইট ওয়ার্ডপ্রেস দিয়ে তৈরি।

যত দিন যেতে থাকে মানুষ তত ওয়েবের ওপরে নির্ভর হতে থাকে। কারণ ওয়েবের মাধ্যমে একজন আর একজনের সাথে খুব সহজেই নিজেকে যুক্ত করতে পারে। একটা ইনফরমেশন যদি ওয়েবে থাকে তাহলে সব ডিভাইস থেকেই সেটা একই সময়ে এবং দ্রুত এক্সেস নেওয়া যায়। ওয়েব তখন একটা সেন্ট্রাল কম্পিউটারের মত কাজ করে যার সাথে বাকি সমস্ত ডিভাইস গুলো কানেক্টেড। মানুষ আস্তে আস্তে বুঝতে পারে যে ওয়েবকে ব্যবহার করে আমরা শুধুমাত্র ইনফরমেশনই শেয়ার করতে পারি না, আরও অনেক কিছুই করা সম্ভব। জন্ম নেয় ওয়েব অ্যাপলিকেশনের। ছোট ছোট যেই কাজ গুলো সফটওয়্যার ইন্সটল করে আমাদের করতে হতো, সেই সমস্ত কাজ গুলো ওয়েবে শিফট হতে শুরু করলো। ব্যাকিং থেকে এন্ট্রোরপ্রাইজ সমস্ত অ্যাপলিকেশনই আস্তে আস্তে ওয়েবের মাধ্যমে ছড়িয়ে যেতে থাকলো। ওয়েব অ্যাপলিকেশনের চাহিদা বৃদ্ধি পাওয়ার প্রথম কারণ হলো এর মাধ্যমে দ্রুত ইউজারের কাছে পৌঁছানো যায়। যার ফলে সোশ্যাল অ্যাপ, গেম সব জায়গাতেই আজকে আমরা ওয়েবের ব্যবহার দেখতে পাই।

ওয়েব অ্যাপলিকেশনের চাহিদা বৃদ্ধি পাওয়ার সাথে সাথে দুইটা বিষয়ের প্রতি ডেভেলপারদের নজর গেল। প্রথমটা হচ্ছে পার্ফরমেন্স আর দ্বিতীয়টা হচ্ছে সিকিউরিটি। একসময় আমরা প্রত্যেকে প্রত্যেকের পিসিতে সফটওয়্যার ইন্সটল করে ব্যবহার করতাম, আর এখন একটা ব্রাউজার ইন্সটল করে প্রায় সব সফটওয়্যারই ইন্টারনেটের মাধ্যমে ব্যবহার করি। একটা মাত্র সার্ভার পিসিতে এই অ্যাপলিকেশন গুলো আছে যা আমাদের সবাইকে সার্ভিস দিচ্ছে। লক্ষ লক্ষ মানুষ যখন একই সময়ে ওই একটা পিসিকেই ব্যবহার করছে তখন কি ওই একটা পিসির পক্ষে সম্ভব এত রিকুয়েষ্ট হ্যান্ডেল করা, এত অপারেশন হ্যান্ডেল করা? অবশ্যই সম্ভব না। এর জন্য দরকার সিস্টেম আর্কিটেকচার, সার্ভার স্কেলিং এর মত বড় বড় সব বিষয়। আর একটা জিনিসও দরকার। আর সেটা হচ্ছে কোডের পার্ফরমেন্স অপটিমাইজেশন। যদি আপনার অ্যাপলিকেশন ব্যবহার কারীর সংখ্যা খুব কম হয় তাহলে আপনি যে কোনো প্রোগ্রামিং ল্যাংগুয়েজ ব্যবহার করেই অ্যাপলিকেশন ডেভেলপ করতে পারেন। কোনো রকম কোনো অপটিমাইজেশনের দরকার হবে না। কিন্তু যদি আপনার অ্যাপলিকেশন একই সময়ে লক্ষ

---

লক্ষ মানুষকে সার্ভ করতে হয় তাহলে কি করবেন? সার্ভারের পরে সার্ভার যুক্ত করতে থাকবেন? নাকি অ্যাপলিকেশনকে অপটিমাইজ করবেন?

ওয়ার্ডপ্রেস ব্যবহার করে আমরা খুব সহজেই যেকোনো ওয়েবসাইট বা বর্তমানে যে কোনো ওয়েব অ্যাপলিকেশনও তৈরি করতে পারি। ইভেন্ট ম্যানেজমেন্ট, লার্নিং অ্যাপ, ইকমার্স সাইট থেকে শুরু করে সোশ্যাল অ্যাপ পর্যন্ত আমরা ওয়ার্ডপ্রেস ব্যবহার করে বানাতে পারি। আর এই কাজ গুলো করার জন্য কোডিং এর দক্ষতাও থাকার দরকার নেই। অল্প স্বল্প যদি পিএইচপি এর জ্ঞান থাকে তাহলে তো আর কোনো কথায় নেই। কিন্তু কখনো চিন্তা করে দেখছেন যে, ওয়ার্ড প্রেস ব্যবহার করে যদি সব এত সহজেই করা যায় তাহলে গুগল, মাইক্রোসফট বা ফেসবুকের মত কোম্পানী গুলো কেন ওয়ার্ড প্রেস ব্যবহার করছে না? কেন তারা কোটি কোটি টাকা খরচ করে, হাজার হাজার ডেভেলপার রেখে তাদের সার্ভিস গুলো বানাচ্ছে? তারা যেই সার্ভিস গুলো প্রোভাইড করে সেগুলোকি ওয়ার্ডপ্রেস দিয়ে বানানো যায় না? ফেসবুক যখন প্রথম ডেভেলপড হয়েছিল, তখন এর ল্যাংগুয়েজ হিসেবে ব্যবহার করা হয়েছিল পিএইচপিকে। কিন্তু কি এমন ঘটলো যে তাদের এই ল্যাংগুয়েজ থেকে সরে এসে নতুন ল্যাংগুয়েজ ডেভেলপ করে নিজের সাইট নতুন করে ডেভেলপ করতে হল? কখনো কি মনে প্রশ্ন জাগে না যে সবাই এত করে ডাটা স্ট্রাকচার অ্যালগোরিদম শিখতে বলে, এগুলোর ব্যবহার আমি ডেভেলপমেন্টে কোথায় করলাম? নাকি ওয়েব ডেভেলপমেন্টে এগুলো লাগে না?

যেকোনো একটা অ্যাপলিকেশন তৈরির পূর্বে আপনাকে জানতে হবে এই অ্যাপলিকেশনটা আপনি কত মানুষের জন্য ডেভেলপ করছেন। এর পরে আপনাকে দেখতে হবে এই অ্যাপলিকেশন কি শুধু ডাটা নিয়েই কাজ করবে নাকি কম্পিউটিং এরও প্রচুর কাজ থাকবে। ১ মিলিয়ন মানুষ ব্যবহার করবে এই রকম একটা ইকমার্স সাইট এবং ভিডিও স্ট্রিমিং সাইট কিন্তু এক না। দুইটা ডেভেলপ করতে সম্পূর্ণ ভিন্ন ভিন্ন টেকনোলজি লাগবে। আবার একই টেকনলজি ব্যবহার করে দুইটা অ্যাপলিকেশনই ডেভেলপ করা যায়। তবে আপনি যদি পিএইচপি, নোডজেএস বা পাইথন ব্যবহার করে স্ট্রীমিং অ্যাপলিকেশন ডেভেলপ করেন তাহলে আপনার অনেক বেশি সার্ভার প্রয়োজন হবে। যখন আপনার ইউজার অনেক বেশি হবে তখন এমন একটা ল্যাংগুয়েজ আপনাকে বেছে নিতে হবে যার এক্সিকিউশন স্পীড অনেক বেশি, কম মেমরিতেও অনেক বেশি কাজ করতে পারে। এখানে আপনি কি কোড লিখছেন সেটাও অনেক গুরুত্বপূর্ণ বিষয়। আর এখানেই কাজে লাগে ডাটা স্ট্রাকচার এবং অ্যালগোরিদম এর থিওরি গুলো। বড় বড় কোম্পানি গুলো কোটি কোটি টাকা স্যালারি দিয়ে

---

অ্যালগোরিদমিষ্ট রাখে এই জন্য যে, যেন তারা প্রতিটা অ্যালগোরিদম এর এক্সিকিউশন টাইমটা কমিয়ে আনতে পারে। প্রতিটা রিকুয়েস্ট থেকে যদি ১ মিলি সেকেন্ড সময় বাঁচানো যায় তাহলে মাস শেষে ফেসবুক গুগলের মত কোম্পানি গুলোর কোটি কোটি টাকা খরচ বেঁচে যাবে।

একটা অ্যাপ্লিকেশন বানানো আর সেই অ্যাপ্লিকেশন দিয়ে ব্যবসা করা মাটেও এক বিষয় না। আপনি যখন শিখবেন তখন আপনি যা খুশি তাই শিখতে পারেন, কিন্তু যখন ব্যবসার উদ্দেশ্যে কোনো অ্যাপ্লিকেশন বানাবেন তখন অবশ্যই আপনি কোন ল্যাংগুয়েজ, কোন ফ্রেমওয়ার্কে কাজ করতে পছন্দ করেন এটা মুখ্য বিষয় না। মুখ্য বিষয় হচ্ছে কোন ল্যাংগুয়েজটা আপনার বিকৃয়ারণেন্টটাকে সব থেকে ভালোভাবে ফুলফিল করতে পারছে। শুধু ডেভেলপমেন্টের সময় না, ডেভেলপমেন্টের পরেও যত দিন অ্যাপ্লিকেশনটা ব্যবহার হবে ততদিন কে সব থেকে ভালো সাপোর্ট দিতে পারবে। আর এই জন্যই গুগল, মাইক্রোসফটের মত কোম্পানি কখনো একটা ল্যাংগুয়েজ ব্যবহার করেনা তাদের সার্ভিস গুলো তৈরির ক্ষেত্রে। যেই ল্যাংগুয়েজটা যেই কাজে ভালো সেই কাজ করার জন্য সেই ল্যাঙ্গুয়েজটাই ব্যবহার করে। স্টার্টআপ আইডিয়া গুলোর ক্ষেত্রে বিষয়টা ভিন্ন, কারণ সেই ক্ষেত্রে আপনি জানেনই না আপনার আইডিয়াটা মানুষ পছন্দ করবে কিনা। এই জন্য দ্রুত প্রোটোটাইপ তৈরি করা যায় এমন একটা ল্যাংগুয়েজ ব্যবহার করেই মার্কেটে নেমে পরা উচিত। পরবর্তীতে মানুষ যদি আপনার অ্যাপ্লিকেশন পছন্দ করে তখন ব্যবহারকারীর সংখ্যা বিবেচনা করে নতুন করে অ্যাপ্লিকেশনটি ডেভেলপ করা যাবে।

ব্যাকেন্ড অ্যাপ্লিকেশন আপনি যে কোন ল্যাংগুয়েজ ব্যবহার করেই ডেভেলপ করতে পারেন। বাজারে প্রচুর ল্যাংগুয়েজ এবং ফ্রেমওয়ার্ক আছে ব্যাকেন্ডের জন্য। কিন্তু আমি এখানে অল্প কয়েকটা ল্যাংগুয়েজ সম্পর্কেই কথা বলব যেগুলো বর্তমানে খুবই জনপ্রিয় এবং যার জব মার্কেটও খুব বড়।

**PHP:** ব্যাকেন্ড ডেভেলপমেন্ট শিখার জন্য এর থেকে সহজ ল্যাংগুয়েজ আর নেই। কারণ এই ল্যাঙ্গুয়েজটা তৈরিই করা হয়েছে ওয়েবের ব্যাকেন্ড নিয়ে কাজ করার জন্য। ওয়েবের আদিপিতা বলা হয় এই ল্যাংগুয়েজকে। যেকোনো ধরনের ওয়েবসাইট ডেভেলপ করার জন্য পিএইচপি এর কোনো জুড়ি নেই। তবে ব্যবহারকারীর সংখ্যা বৃদ্ধি পেলে পিএইচপি তখন আর পেরে ওঠে না। অবশ্যই আপনি সার্ভার স্কেল করার মাধ্যমে সাপোর্ট বজায় রাখতে পারেন, কিন্তু এতে আপনার প্রচুর অর্থ অযথায় খরচ হবে। তাই পার্সোনাল ওয়েবসাইট বা

---

ইনফরমেশন শেয়ার করবে এরকম ওয়েব সাইট তৈরি করার জন্যই পিএইচপি বেঁচে নেওয়া উচিত।

ওয়েব অ্যাপলিকেশন ডেভেলপমেন্টের দুইটা আর্কিটেকচার খুবই জনপ্রিয়। একটা হচ্ছে মনোলিথিক যা অনেক পুরাতন একটা আর্কিটেকচার আর একটা হচ্ছে মাইক্রো সার্ভিস আর্কিটেকচার। মনোলিথিক আর্কিটেকচারে একটা অ্যাপলিকেশন যত বড়ই হোক না কেন সমস্ত কোড বেস এক জায়গাতেই থাকে। যার ফলে এই আর্কিটেকচার ব্যবহার করে একটা অ্যাপলিকেশন শুরু করা অনেক সহজ। কিন্তু আপনার অ্যাপলিকেশনটা যদি এমন হয় যে, এখানে প্রতিনিয়ত নতুন নতুন ফিচার ইমপ্লিমেন্ট হচ্ছে তাহলে অন্ন কিছুদিনের ভিতরেই আপনার অ্যাপলিকেশন ম্যানেজ করা প্রায় অসম্ভব হয়ে যাবে। একটা ছোট বাগ ফিক্সিং এর জন্যও আপনার অনেক প্যারা সহ্য করতে হবে। আস্তে আস্তে টিম বড় হতে থাকবে এবং ডেভেলপমেন্ট প্রোসেস, নতুন আপডেট সব কিছু স্লো হতে থাকবে। এই সমস্যার কারণে বর্তমানে প্রায় সব অ্যাপলিকেশনই মাইক্রো সার্ভিস আর্কিটেকচার ফলো করে তৈরি করা হয়। এই ক্ষেত্রে পুরো অ্যাপ্লিকেশনটাকে ছোট ছোট অসংখ্য সার্ভিসে ভাগ করা হয়। একটা সার্ভিস একটা নির্দিষ্ট কাজ, একটা নির্দিষ্ট অ্যাপ। এই অ্যাপটা ম্যানেজ করা হবে ইন্ডিভিজুয়ালি, টেষ্ট করা হবে ইন্ডিভিজুয়ালি এবং ডেপলয় করা হবে ইন্ডিভিজুয়ালি। একটা অ্যাপলিকেশন থাকবে যার নাম এপিআই গেটওয়ে, যে সমস্ত ছোট ছোট অ্যাপলিকেশন গুলোকে একত্র করবে এবং ইউজারের সাথে ইন্টারেক্ট করবে। এই ধরনের মাইক্রোসার্ভিস বেসড অ্যাপলিকেশন গুলো তৈরি করার শুরুর দিকে তো আপনাকে অনেক পেইন নিতে হবে, কিন্তু পরবর্তীতে সব কিছু এতটাই স্মৃথিলি চলবে, টাইম টু টাইম আপডেট, ফাস্ট বাগ ফিক্সিং সব কিছুই হবে, যে আপনার ইউজার বুঝেই উঠতে পারবে না বিহাইন্ড দ্যা সিন কি ঘটছে।

পিএইচপি ব্যবহার করেও মাইক্রোসার্ভিস আর্কিটেকচার তৈরি করা যায়, তবে এর কম্পাইলের কিছু ক্রটির কারণে অন্যান্য ল্যাংগুয়েজ ব্যবহার করে যতটা ভালোভাবে মাইক্রোসার্ভিস তৈরি করা যায়, পিএইচপিতে এতটা ভালো ভাবে করা যায় না। আর পিএইচপির বেশিরভাগ জনপ্রিয় ফ্রেমওয়ার্কই মনোলিথিক আর্কিটেকচার মডেলে তৈরি করা। তবে আপনারা যারা পিএইচপি ডেভেলপার আছেন তাদের জন্য সুখবর হচ্ছে পিএইচপি ৮ এ আপনারা JIT (Just In Time) কম্পাইলার পেতে যাচ্ছেন পিএইচপি এর সাথে। যার ফলে আর দশটা মডার্ন ল্যাংগুয়েজের মতই পিএইচপি কাজ করতে যাচ্ছে।

---

পিএইচপি এর জনপ্রিয় ফ্রেমওয়ার্কের কথা বললে সবার প্রথমে বলতে হবে লারাভেল এর নাম। আমি পার্সোনালি পিএইচপি নিয়ে খুব কম কাজ করি এবং পিএইচপি এর ফ্রেমওয়ার্ক বলতে এই একটা মাত্র ফ্রেমওয়ার্ক সম্পর্কেই কিছুটা জ্ঞান রাখি। র্যাপিড অ্যাপলিকেশন ডেভেলপ করার জন্য খুব সুন্দর একটা ফ্রেমওয়ার্ক এটি। একজন পিএইচপি ডেভেলপার হিসেবে আমি মনে করি সবার এই ফ্রেমওয়ার্কটা শেখা উচিত।

অনেকেই মনে করেন ওয়ার্ড্রেস পিএইচপি এর একটা ফ্রেমওয়ার্ক। ওয়ার্ড্রেস এর থিম ডেভেলপ করার অর্থ হচ্ছে ওয়েব ডেভেলপ করা। শুনতে একটু খারাপ শোনা গেলেও বলি, আপনি ওয়েব ডেভেলপমেন্ট করছেন না। ওয়ার্ড্রেস কোনো ফ্রেমওয়ার্ক না, এটা একটা কন্টেন্ট ম্যানেজমেন্ট সিস্টেম। যেখানে বসে আপনি আপনার ওয়েবসাইটের চেহারা কেমন হবে, ওয়েবসাইটে কি কি কন্টেন্ট থাকবে তা ম্যানেজ করতে পারেন। ওয়ার্ড্রেস এর সাথে একটা ড্যাসবোর্ডও দিয়েছে, যার ফলে আপনার অ্যাপলিকেশনের অন্যান্য কিছু কাজও এখানে বসেই করা যায়। কিন্তু কোনো ভাবেই এটা কোনো ওয়েব ফ্রেমওয়ার্ক না। আপনি যদি ওয়ার্ড্রেসের জন্য প্লাগিন তৈরি করেন আপনি ওয়ার্ড্রেস প্লাগিন ডেভেলপার, আর যদি থিম তৈরি করেন তাহলে ওয়ার্ড্রেস থিম ডেভেলপার যার কোনোটার সাথেই ওয়েব অ্যাপলিকেশন ডেভেলপমেন্টের কোনো সম্পর্ক নেই। আপনি যদি লারাভেল ব্যবহার করেন, অথবা কোর পিএইচপি ব্যবহার করে একটা ওয়েবসাইট ডেভেলপ করেন, আপনি অবশ্যই একজন ওয়েব ডেভেলপার।

**NodeJS & Javascript:** আমাদের দেশের জন্য খুবই নতুন একটা টেকনোলজি হচ্ছে NodeJS, যা মূলত একটা রানটাইম। আপনি একে জাভাস্ক্রিপ্টের কম্পাইলার ভাবতে পারেন। যা ব্যবহার করে আমরা জাভাস্ক্রিপ্ট কোড আমাদের পিসিতে রান করতে পারি। মডার্ন ওয়েব ডেভেলপমেন্টের জন্য নোডজেএস এখন ডেভেলপার দের প্রথম পছন্দ। ২০১১ সাল থেকে সফল ভাবে নোডজেএস তার বিস্তৃতি লাভ করছে এবং সবার মন জয় করছে। বড় বড় কোম্পানী গুলো এখন প্রচুর কাজে নোডজেএস ব্যবহার করছে। আজকে আমরা জাভাস্ক্রিপ্টের যেই জয়গান শুনি তার পিছনের যে কারিগর, সে হল নোডজেএস।

নোডজেএস ব্যবহার করে পার্সোনাল ওয়েবসাইট থেকে শুরু করে ইকমার্স সাইট, ভিডিও স্ট্রিমিং সাইট, গেম সার্ভার সহ যেকোনো ধরনের ওয়েবসাইট বা অ্যাপলিকেশন তৈরি করা যায়। তবে শুধুমাত্র ডাটা ইন্টেলিজিভ অ্যাপলিকেশনগুলো নোড ব্যবহার করে ডেভেলপ করা হলে আপনি লাভবান হবেন। যেই সব অ্যাপলিকেশন গুলোতে প্রচুর সিপিইউ এর কাজ থাকে

---

সেই সব অ্যাপলিকেশন তৈরিতে নোডজেএস ব্যবহার না করাই ভালো, কারণ নোডজেএস সিঙ্গেল থ্রেডেড একটা প্লাটফর্ম। নোডজেএস ব্যবহার করে খুব সহজে খুবই দ্রুত গতি সম্পন্ন ওয়েবসাইট তৈরি করা যায়। কিন্তু ওয়ার্ডপ্রেসের মত কোনো সিএমএস না থাকার কারণে মানুষ ওয়েবসাইট তৈরিতে পিএইচপিকেই ভালোবাসে। ওয়ার্ডপ্রেসের মত জনপ্রিয় সিএমএস না থাকলেও নোডজেএস এর আছে উজনখানেক পাওয়ারফুল সিএমএস এবং ফ্রেমওয়ার্ক। তবে নোডজেএস এর ব্যবহার ওয়েবসাইটের থেকে ওয়েব অ্যাপলিকেশন তৈরি করতেই বেশি দেখা যায়। নোডজেএস সিঙ্গেল থ্রেড এবং নন ব্লকিং হওয়ায় অন্যান্য ল্যাংগুয়েজের থেকে অনেক দ্রুত এবং বেশি ইউজারকে একই সময়ে সার্ভিস প্রদান করতে পারে। যার ফলে অনেক ছোট সার্ভার ব্যবহার করেও একই সময়ে অনেক বেশি ইউজার রিকুয়েস্ট হ্যান্ডেল করা সম্ভব হয়। নোডজেএস এর জনপ্রিয়তার একটা প্রধান কারণ হচ্ছে এটি।

বড় বড় কোম্পানি গুলো মাইক্রোসার্ভিসের দিকে ঝুঁকে পরার কারণে সব থেকে বেশি লাভবান হয়েছে নোডজেএস। যেহেতু নোডজেএস নন ব্লকিং তাই এপিআই গেটওয়ে তৈরি করার জন্য এর জুড়ি নেই। সিপিইউ ইন্টেন্সিভ কাজ গুলো - জাভা, সিশার্প বা গোল্যাং এর মত ল্যাংগুয়েজ দিয়ে হ্যান্ডেল করে, ডাটা ইন্টেন্সিভ কাজ গুলো করা হচ্ছে নোডজেএস ব্যবহার করে। যার কারণে কোম্পানি গুলো অর্থনৈতিক ভাবে লাভবান হচ্ছেন। মাইক্রোসার্ভিস তৈরির জন্য নোডজেএস এর রয়েছে অসংখ্য ফ্রেমওয়ার্ক যা ব্যবহার করে খুব অল্প সময়েই একটা মাইক্রোসার্ভিস ডেভেলপ করা যায়।

নোডজেএস এর যে শুধু ভালো দিকই রয়েছে কোনো খারাপ দিক নেই এমনটা না। এর সব থেকে বাজে বিষয় হচ্ছে এটি সিঙ্গেল থ্রেডেড, যদিও এখন চাইল্ড প্রোসেস এবং ওয়ার্কার থ্রেড ব্যবহার করে এই সমস্যাটার সমাধান অনেকাংশেই করা যায় তারপরেও যেই কাজ গুলো করতে প্রচুর সিপিইউ পাওয়ার দরকার হয় সেই কাজ গুলো করার জন্য নোডজেএস মোটেও ভালো কোনো সমাধান না। নোডজেএস নতুন একটা টেকনোলজি। আমি বলব না যে এটা পুরোপুরি ম্যাচুউর হয়ে গেছে। এর ম্যাচুউর হয়ে উঠতে, সব রকম টুলিং সাপোর্ট দিতে আরও সময় লাগবে। তবে যেহেতু বড় বড় কোম্পানী গুলো তাদের বড় বড় প্রজেক্টে এটা ব্যবহার করছে তাই আমি মনে করি এটা একটা ভালো সমাধান হতে পারে।

একজন ফ্রন্টেন্ড ডেভেলপারের জন্য আমার মনে হয় না যে নোডজেএস এর থেকে ভালো কোনো সমাধান আছে। কারণ এখানে আপনাকে ব্যবহার করতে হবে জাভাস্ক্রিপ্ট। আর একজন ফ্রন্টেন্ড ডেভেলপার হিসেবে আপনাকে সব সময় জাভাস্ক্রিপ্ট নিয়েই কাজ করতে

---

হয়। তো আপনি যদি এখন ব্যাকেন্ড ডেভেলপমেন্ট করতে চান তাহলে নতুন করে কোনো ল্যাংগুয়েজ আপনাকে শিখতে হল না। শুধুমাত্র ব্যাকেন্ড ডেভেলপমেন্টের টেকনিক এবং একটা ভালো ব্যাকেন্ড ফ্রেমওয়ার্ক শিখে নিলেই হল। ফ্রেমওয়ার্কের কথা বললে সবার প্রথমে যেই নামটা উঠে আসে তা হল ExpressJS. নোডজেএস এর সব থেকে জনপ্রিয় মাইক্রো ফ্রেমওয়ার্ক যা ব্যবহার করে আপনি পিএইচপি এর মত ওয়েবসাইটও তৈরি করতে পারবেন আবার পেপালের মত পেমেন্ট গেটওয়েও তৈরি করতে পারবেন। মাইক্রো ফ্রেমওয়ার্ক হওয়ার কারণে মনের মত কাস্টমাইজ করে কাজ করা যায়। মাইক্রোসার্ভিস তৈরি করার জন্যও এটা খুব সুন্দর একটা সমাধান।

**Python:** বর্তমান বিশ্বের সব থেকে জনপ্রিয় এবং বহুল ব্যবহৃত প্রোগ্রামিং ল্যাংগুয়েজের নাম হচ্ছে পাইথন। জেনারেল পার্স এই ল্যাঙ্গুয়েজটা দিয়ে করা যায় না বলতে আমার দেখা শুধু একটা বিষয়ই আছে আর তাহলো ফন্টেন্ড ডেভেলপমেন্ট। বাকি সব কাজ সব থেকে সহজে, সব থেকে কম কোড লিখে করা যায় পাইথন ব্যবহার করে। বর্তমানে ডাটা সাইন্স এবং রিসার্চের কাজেই পাইথন সব থেকে বেশি ব্যবহৃত হয়। তবে ওয়েব ডেভেলপমেন্টের ক্ষেত্রেও পিছিয়ে নেই পাইথন। র্যাপিড ডেভেলপমেন্টের জন্য এর রয়েছে বিশাল এক ফ্রেমওয়ার্ক যার নাম জ্যাংগো। তবে জ্যাংগো মনোলিথিক হওয়াতে খুব বড় অ্যাপলিকেশন এর কাজে ব্যবহার না করাই ভালো। তবে আপনি লারাভেল ব্যবহার করে যেই অ্যাপলিকেশন বানাতেন সেই একই অ্যাপলিকেশন যদি আপনি জ্যাংগো ব্যবহার করে বানান, তাহলে নিঃসন্দেহে আপনার অ্যাপলিকেশনের পার্ফরমেন্স আগের থেকে অনেক বেটার হবে। অ্যাপলিকেশনের পার্ফরমেন্স বেশির ভাগ ক্ষেত্রেই নির্ভর করে ল্যাংগুয়েজের ওপরে। ফ্রেমওয়ার্কের ওপরেও নির্ভর করে, ফ্রেমওয়ার্ক গুলো যদি ভালো ভাবে তাদের কোড অপটিমাইজ করে তাহলে পার্ফরমেন্স অনেক বেটার হয়। এই ক্ষেত্রে লারাভেল অনেক ভালো কাজ করলেও পিএইচপি এর জন্য পেরে ওঠেনি। পিএইচপি পাইথনের থেকে অনেক বেশি ল্লো একটা ল্যাংগুয়েজ।

তবে আপনি যদি ভেবে থাকেন, পাইথন তো ফাস্টার, পাইথন ব্যবহার করে অনেক বড় বড় সিপিইউ ইন্টেন্সিভ অ্যাপলিকেশন বানাবো তাহলে একটু ভুল হবে। পাইথন পিএইচপি এর থেকে ফাস্টার কিন্তু বাস্তব জীবনে পাইথন খুবই ল্লো একটা ল্যাংগুয়েজ। কারণ এটা ইন্টারপ্রেটেড ল্যাংগুয়েজ। অনেকেই নোডজেএস এর সাথে পাইথনের তুলনা দিয়ে থাকেন। অনেকেই মনে করেন পাইথন নোডজেএস এর থেকে ফাস্ট, কিন্তু দুঃখজনক হলেও সত নোডজেএস এখন অবধি সব থেকে ফাস্টেষ্ট ব্যাকেন্ড সল্যুশন। আমি জানি জাভাস্ক্রিপ্ট সব

---

থেকে ফাস্ট ল্যাংগুয়েজ না এবং সব ক্ষেত্রে নোডজেএস ফাস্টেষ্ট সলুশনও না, তবে বেশিরভাগ ক্ষেত্রেই এটা সব থেকে ফাস্ট। নোডজেএস এত ফাস্ট এক্রিকিউট হওয়ার পিছনে দুইটা কারণ। প্রথম কারণ হচ্ছে গুগলের V8 ইঞ্জিন এবং দ্বিতীয় কারণ হচ্ছে এটা নন ব্লকিং। জাভা, সিশার্পের মত ল্যাংগুয়েজ গুলোর এক্রিকিউশন টাইম জাভাস্ক্রিপ্টের থেকে অনেক অনেক কম, এরা অনেক দ্রুত কোড এক্রিকিউট করে থাকে। কিন্তু ওয়েবের ক্ষেত্রে এরা রিকুয়েষ্ট ব্লক করে কাজ করে, আর নোডজেএস ব্লক করে না। তাই নোডজেএস এর থেকে ফাস্ট ব্যাকেন্ড সলুশন এখন অবধি বাজারে নেই।

র্যাপিড অ্যাপলিকেশন ডেভেলপমেন্টের জন্য আপনি পিএইচপি এর বদলে পাইথন, লারাভেল এর বদলে জ্যাংগো বেঁচে নিতে পারেন। তবে এই ক্ষেত্রেও মাথায় রাখতে হবে যদি আপনার ইউজার বেস অনেক বেশি হয়ে যায় সেইক্ষেত্রে কিন্তু আপনি প্রক্লেমের সম্মুখীন হবেন। স্টার্টাপ কোম্পানি গুলোর প্রথম চয়েস পাইথন। কারণ খুব দ্রুত বাজারে আসতে চাইলে আপনাকে জ্যাংগোর কাছেই আসতে হবে। যদি আপনি পিএইচপি শিখতে না চান তাহলে আমার প্রথম রিকমেন্ডেশন থাকবে পাইথন। আর যদি আপনার জাভাস্ক্রিপ্ট জানা থাকে তাহলে আমি বলবো নোডজেএস। তবে ওয়েব ডেভেলপমেন্ট বাদে যদি আপনি চিন্তা করেন তাহলে আমি বলব পাইথন শিখুন। আমি সবাইকে নূন্যতম তিনটা ল্যাংগুয়েজ শেখার সাজেশন দিয়ে থাকি। প্রথমটা হচ্ছে সি, দ্বিতীয়টা হচ্ছে জাভাস্ক্রিপ্ট আর তৃতীয়টা হচ্ছে পাইথন। যদি কারোর সময় কম থাকে, আর দ্রুত ক্যারিয়ার গড়তে চাই তাদের জন্য আমার এই সাজেশনটা।

**Java:** এন্টারপ্রাইজ অ্যাপলিকেশনের কথা আসলেই প্রথম যে নামটা সবার মনে আসে তা হলো জাভা। বহু বছর ধরে জাভা সব থেকে বড় বড় অ্যাপলিকেশন ডেভেলপ করে আসছে। যেখানে পার্ফরমেন্সের দরকার হয়, যেখানে সিকুরিটি দরকার হয় সেখানেই জাভা। ছোটোখাটো কোনো কাজের জন্য জাভার নাম মনেও আনবেন না, তাহলে জাভাকে অসম্মান করা হবে আর আপনার সময়, আপনার অর্থের বারোটা বাজবে। প্রথম থেকেই বলে আসছি পিএইচপি, পাইথন, নোডজেএস ব্যবহার করে সিপিইউ ইন্টেন্সিভ কাজ গুলো করবেন না। তাহলে করবো কি দিয়ে? সিপিইউ ইন্টেন্সিভ কাজ বলতে আসলে বোঝায় যেই কাজ গুলো করতে প্রচুর সিপিইউ পাওয়ার লাগবে, মাল্টিপল থ্রেড লাগবে। এই কাজ গুলো এমন ল্যাংগুয়েজ ব্যবহার করে করা উচিত যা খুব দ্রুত কম মেমোরি খরচ করে এক্রিকিউট হতে পারে। জাভা এর উৎকৃষ্ট একটা উদাহরণ। জাভার থেকেও ফাস্ট প্রোগ্রামিং ল্যাংগুয়েজ বাজারে আছে, কিন্তু ওয়েবের জন্য জাভার মত সাপোর্ট এবং এক্রিকিউশন পাওয়ার আর কোনো

---

ল্যাংগুয়েজের নেই। তাই সিপিইউ ইন্টেন্সিভ কাজের জন্য জাভাকে ইন্ডারস্ট্রি স্ট্যান্ডার্ড ধরে নিতে পারেন।

জাভা ব্যবহার করা হয় নেটফ্রিক্সের মত সার্ভিস তৈরির কাজে যেখানে একই সময়ে লক্ষ লক্ষ মানুষের কাছে ভিডিও স্ট্রিম সার্ভিস পৌঁছে দেওয়া হচ্ছে। জাভা ব্যবহার করা হয় IntelliJ Idea, Netbeans, Eclipse এর মত অ্যাডভান্সড ও পাওয়ারফুল IDE তৈরির কাজে। জাভা ব্যবহার করা হয় ERP এবং ব্যাংকিং সফটওয়্যার তৈরির কাজে। ওয়েবে জাভার বহুবিধ ব্যবহার রয়েছে কিন্তু তার কোনটাই ছোটখাটো কাজের জন্য না। জাভার সব থেকে জনপ্রিয় ওয়েব ফ্রেমওয়ার্ক হচ্ছে Spring যা ব্যবহার করে আপনি মনোলিথিক এবং মাইক্রোসার্ভিস দুটো আর্কিটেকচারেই অ্যাপ্লিকেশন তৈরি করতে পারবেন। Spring ফ্রেমওয়ার্কটার অসংখ্য মডিউল আছে, একেকটা মডিউল একেকটা কাজে ব্যবহৃত হয়।

**ASP.NET & C#:** র্যাপিড এন্টারপ্রাইজ অ্যাপ্লিকেশন তৈরির অন্য আর একটা সল্যুশন হচ্ছে ASP.NET যা জাভা এর থেকে সামান্য পরিমাণ ম্লো। আপনি যেই কাজ গুলোতে জাভা ব্যবহার করতেন ঠিক একই কাজে সিশার্প এবং ASP.NET ব্যবহার করা যায়। C# হচ্ছে একটা প্রোগ্রামিং ল্যাংগুয়েজ যা মাইক্রোসফটের তৈরি এবং অনেকটা জাভার মতই, তবে কিছু কিছু ক্ষেত্রে জাভার থেকে এর ইমপ্লিমেন্টেশনটা আমার কাছে বেশি ভালো মনে হয়েছে। আর ASP.NET হচ্ছে ফ্রেমওয়ার্ক যা MVC প্যাটার্ন কাজ করে থাকে। এই ফ্রেমওয়ার্ক ব্যবহার করে আপনি খুব কম সময়েই একটা বড় সড় অ্যাপ্লিকেশন তৈরি করতে পারবেন। এবং পার্ফরমেন্সও খুব ভালো হবে। আমাদের দেশে শুরু থেকেই ASP.NET এর চাহিদা ছিল, এখনো আছে।

**Special Mention:** ব্যাকেন্ডের ল্যাংগুয়েজের আসলে শেষ নেই, যত বলব ততই বলা যাবে। তবে আমি এখানে সমসাময়িক ব্যবহৃত সব থেকে জনপ্রিয় ল্যাংগুয়েজ গুলোর কথায় উল্লেখ করলাম। তবে আরও কয়েকটা ল্যাংগুয়েজ এবং প্লাটফর্ম সম্পর্কে না বললেই নয়, যেগুলো এর মধ্যেই বাজারে চলে এসেছে। ডেভেলপমেন্টের কাজে ব্যবহৃতও হচ্ছে, মানুষের ভালোবাসাও পাচ্ছে। তবে এখনো পুরোপুরি ভাবে নিজের অবস্থান পরিষ্কার করতে পারেনি। সবার প্রথমে যার নাম বলতে হয় সেটা হচ্ছে Golang. গুগলের ডেভেলপ করা একটা প্রোগ্রামিং ল্যাংগুয়েজ যা কিনা সি, সি++ এর মতই ফাংশ্ব, লো লেভেল, লো এক্সিকিউশন টাইম এবং অনেক সহজ। মাইক্রোসার্ভিস তৈরিতে গোল্যাং এবং কোন জুড়ি নেই। অন্ন কিছু দিনের ভিতরেই আমরা হ্যাত টপ লিস্টে গোল্যাং কে দেখতে পারবো। কারণ

---

এটা লোলেভেল ল্যাংগুয়েজ হলেও খুব সহজেই এটা ব্যবহার করে একটা ওয়েব অ্যাপ্লিকেশন তৈরি করা যায়।

গোল্যাং এর মত প্রায় সম আর একটা ল্যাংগুয়েজ হচ্ছে Rust যা মজিলা ডেভেলপ করেছে। এটাও অনেক লো লেভেল কিন্তু হাই লেভেল সিনট্যাক্স। যার ফলে রাষ্ট্র ব্যবহৃত হচ্ছে প্রচুর কাজে। আপনারা হয়ত ফ্রন্টেন্ড ওয়েব অ্যাসেম্বলি এর নাম শুনেছেন। রাষ্ট্র ব্যবহার করে খুব সহজেই আপনি ওয়েব অ্যাসেম্বলি এর কাজ করতে পারবেন। তাহলে বলা যায় রাষ্ট্র ব্যবহার করে ফ্রন্টেন্ড ব্যাকেন্ড দুইটা কাজই করা যাবে জাভাস্ক্রিপ্টের মত। আপনারা হয়তো জনপ্রিয় প্রোটোটাইপিং অ্যাপস Figma এর নাম শুনেছেন যা তৈরি করা হয়েছে রাষ্ট্র এবং ওয়েব অ্যাসেম্বলি ব্যবহার করে।

সব শেষে যার কথা বলবো সে হল একটা রানটাইম। নোডজেএস এর মতোই জাভাস্ক্রিপ্ট, টাইপস্ক্রিপ্টের একটা রানটাইম। নোডজেএস তৈরিতে যে ভুল গুলো করেছিলেন Ryan Dehl, নোডজেএস এ যেই বিষয় গুলো না থাকলে আর যেই ফিচার গুলো থাকলে ভালো হতে বলে তিনি মনে করেন, সেই বিষয় গুলো নিয়েই দীর্ঘদিন ধরে নতুন একটা রান টাইম তৈরি করছেন নোডজেএস এর জনক Ryan Dehl যার নাম Deno. এর মধ্যেই আমরা প্রিরিলিজ ভার্সন দেখেছি এবং দেখেই মুঞ্চ হয়েছি।

আমি প্রথমেই বলেছি ব্যাকেন্ড ডেভেলপমেন্টে আমাদের অনেক বেশি ফ্লেক্সিবিলিটি আছে, কারণ এখানে অসংখ্য অপশন আছে। এক্যাস্ট্র কোন টুলস্টা আপনার কাজে লাগবে সেটা আপনাকেই খুঁজে বের করতে হবে। আপনার যদি পিএইচপি ব্যবহার করেই কাজ হয়ে যায় তাহলে আপনি পিএইচপি ব্যবহার করবেন। যদি কোথাও নোড লাগে তাহলে নোড, রাষ্ট্র লাগলে রাষ্ট্র আর জাভা লাগলে জাভা। কখনো কখনো একটা ল্যাংগুয়েজ দিয়েই আপনার পুরো অ্যাপ্লিকেশন ডেভেলপ করা সম্ভব হবে আবার কখনো কখনো আপনার একাধিক ল্যাংগুয়েজ লাগবে। যেমন নেটফ্রিক্সের কথায় চিন্তা করা যাক। এখানে আমরা যেই ভিডিও স্ট্রিমিং দেখি তার ব্যাকেন্ড আছে জাভা, নেটফ্রিক্স আমাদের ইন্টারেক্টের ওপরে ভিত্তি করে যেই সাজেশন প্রোভাইড করে থাকে তার পিছনে আছে পাইথন আর যার মাধ্যমে ব্যাকেন্ড আর ফ্রন্টেন্ড কানেক্ট হচ্ছে মানে এপিআই তার পিছনে আছে নোডজেএস।

---

একটা পার্সোনাল সাইট তৈরি করা, প্রাকটিসের জন্য একটা অ্যাপ ডেভেলপ করা আর একটা অ্যাপ তৈরি করে তার থেকে ব্যবসা করা সম্পূর্ণ ভিন্ন। আপনারা সবাই অ্যাপ তৈরি করে ব্যবসা করবেন না, হয়ত যারা এই বইটা পড়ছেন তার ভিতরে ১-২% মানুষের মনে এই রকম ইচ্ছে রয়েছে। বাকিরা কি করবেন? হয় ফ্রিলান্সিং করবেন না হয় জব করবেন। যায়ই করেন, বিশ্ব যেভাবে চলছে তার বাইরে তো আপনি চলতে পারবেন না। বিশ্ব যদিকে আগাছে তার বাইরে তো আপনি আগাতে পারবেন না। আপনাকে বিশ্বের সাথে তাল মিলিয়ে চলতে হবে। আর এর জন্য আপনাকে একটা প্রোসেস সম্পর্কে জানতে হবে। নিজের দক্ষতা বৃদ্ধি করেই যেতে হবে। টেকনোলজির এই জগতে কেউ আমরা সেফ না, আজকে আমি যেই টুলস নিয়ে কাজ করছি কালকে সেটা ওল্ড ফ্যাশান হয়ে যেতে পারে। তাই নিজেকে সব সময় শেখার ভিতরে রাখতে হবে। যেন কালকের টেকনোলজির সাথে খাপ খাইয়ে নিতে আমাদের কোনো সমস্যা না হয়। আর একটা বিষয়, টেকনোলজি জগতে ঢিকে থাকতে হলে ইমোশনকে দূরে রাখতে হবে। আমাদের একটাই ইমোশন, আমরা টেকনোলজি ভালোবাসি। স্পেসিফিক কোনো টেকনোলজির প্রেমে হাবুড়ুবু খাওয়া যাবে না। কারণ আপনি একটা টেকনোলজিকে যতই ভালোবাসেন না কেন, সেই টেকনোলজি কিন্তু আপনাকে ছ্যাকা দিতে দুইবার ভাববে না।

# PHP শিখার রেফারেন্স

## Must Read PHP Books:

- PHP in Action: Objects, Design, Agility By Dagnn Reiersol
- Programming PHP: Creating Dynamic Web Pages By Kevin Tatroe
- PHP and MySQL Web Development By Laura Thompson
- PHP & MySQL Novice to Ninja By Tom Butler
- Head First PHP & MySQL By Lynn Beighley
- PHP Objects, Patterns and Practice By Matt Zandstra

## Youtube Channels to Learn PHP

- [Learn with Hasin Hayder](#) (Bangla)
- [Training with Live Projects](#) (Bangla)
- [Learn Hunter](#) (Bangla)
- [Dev Marketer](#) (English)
- [The New Boston](#) (English)
- [The Net Ninja](#) (English)

## Websites to Learn PHP

- [PHP Manual](#)
- [PHP T Point](#)
- [Learn PHP](#)
- [PHP The Right Way](#)
- [Killer PHP](#)

## Training Program for PHP

- [Learn with Hasin Hyder Courses](#)
- [Learn Hunter Live](#)

# Python শিখার রেফারেন্স

## Must Read Python:

- পাইথন পরিচিতি By Tamim Shahriar Subeen
- সহজ ভাষায় পাইথন By মাকসুদুর রহমান মাটিন
- Python Crash Course By Eric Matthews
- Head First Python By Paul Berry
- Python Programming: An Introduction to Computer Science By John Zelle
- Fluent Python: Clear, Concise and Effective Programming By Luciano Ramalho
- Programming Python: Powerful Object Oriented Programming By Mark Lutz

## Youtube Channels to Learn Python

- [Tamim Shahriar](#) (Bangla)
- [Anisul Islam](#) (Bangla)
- [Stack Learner](#) (Bangla) [Upcoming]
- [Corey Schafer](#) (English)
- [Telusko Learning](#) (English)
- [Programming Knowledge](#) (English)

## Websites to Learn Python

- [Python Bangla](#)
- [সহজ ভাষায় পাইথন](#)
- [Google's Python Class](#)
- [Programmiz](#)
- [Real Python](#)

## Training Program for Python

- [Stack Learner Premium Courses](#) (Upcoming)
- [Stack Learner Offline Bootcamps](#) (Upcoming)



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>



---

অধ্যায় নং

## মেরুদণ্ডের কাঠামো

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ ব্যাকেন্ড ডেভেলপমেন্ট
- ✓ ব্লকিং vs নন ব্লকিং
- ✓ ব্যাকেন্ড কি কি শিখতে হবে

---

ব্যাকেন্ড ডেভেলপমেন্টের জন্য আমাদের সামনে উজনখানেক অপশন থাকলেও শুরু করার জন্য আমার মতে নোডজেএস সব থেকে সহজ সমাধান। নোডজেএস শিখলে অন্ততপক্ষে খুব অল্প সময়েই নিজেকে ফুলস্ট্যাক ডেভেলপার হিসেবে দাবী করা যায়। ব্যাকেন্ড ডেভেলপমেন্ট আপনি যখন খুশি তখনই শুরু করতে পারেন। তবে শুরু করার পূর্বে ওয়েব সম্পর্কে নূন্যতম ধারণা আপনাকে অর্জন করে নিতেই হবে। ওয়েব কিভাবে কাজ করে, কিভাবে ওয়েব ডিজাইন করতে হয়, HTML, CSS আর অল্প স্বল্প জাভাস্ক্রিপ্ট শিখে না আসলে ব্যাকেন্ড ডেভেলপমেন্টের আগা মাথা বুঝতেই অনেক সময় লেগে যায়। আর যেহেতু এই সব জানতে জানতে কিছুটা হলেও জাভাস্ক্রিপ্ট সম্পর্কে জ্ঞান অর্জন হয়েই যায়, তাই নতুন করে আবার ল্যাংগুয়েজ শেখার পিছনে সময় দিয়ে লাভ কি, নোডজেএস দিয়েই শুরু করুন। আর আপনি যদি ফ্রন্টেন্ড ডেভেলপার হন, সেই ক্ষেত্রে তো আর কোনো কথায় নেই।

আমাদের দেশে নোডজেএস এখনো একটা নতুন টেকনোলজি। তাই হরহামেশায় নোডজেএসকে জাভাস্ক্রিপ্টের ফ্রেমওয়ার্ক মনে করে ভুল করতে দেখা যায়। নোডজেএস জাভাস্ক্রিপ্টের কোনো ফ্রেমওয়ার্ক না, এটা একটা জাভাস্ক্রিপ্ট রানটাইম। কয়েক বছর আগে জাভাস্ক্রিপ্ট কোড ব্রাউজারে HTML এর সাথে রান করতে হতো, এর বাইরে কোনো ভাবেই রান করা যেত না। কারণ জাভাস্ক্রিপ্টের স্ট্যান্ডএলোন কোনো কম্পাইলার বা রানটাইম ছিল না। জাভাস্ক্রিপ্টকে সবাই খেলনা ল্যাংগুয়েজ হিসেবেই ধারণা করতো, তাই কেউ কখনো এই ল্যাংগুয়েজটাকে ব্রাউজারের বাইরে রান করাতেও চাইনি। ২০০৯ সালে প্রথম এই নিয়ে কাজ শুরু করে Ryan Dahl। কাজ শুরু করার পরে অনেক তাচ্ছিল্যের স্বীকারণ হন তিনি, কারণ তখনও গোটা বিশ্ব জাভাস্ক্রিপ্টের ক্ষমতা সম্পর্কে অবগত ছিল না। আমরা গুগল ক্রোমে জাভাস্ক্রিপ্টের যেই ইঞ্জিন ব্যবহার করি, V8 ইঞ্জিন, সেই একই ইঞ্জিন ব্যবহার করে, একটু মডিফাই করে সাথে ডেক্সটপে চালানোর জন্য কিছু লাইব্রেরী সাপোর্ট দিয়ে একটা রানটাইম অ্যাপলিকেশন তৈরি করলো। আর এটাই হলো নোডজেএস।

নোডজেএস শুধুমাত্র একটি কাজই করে থাকে। আর তাহল জাভাস্ক্রিপ্টকে ব্রাউজারের বাইরে রান করে থাকে। যখন জাভাস্ক্রিপ্ট ব্রাউজারের বাইরে একটা সাধারণ প্রোগ্রামিং ল্যাংগুয়েজ এর মতো কাজ করবে তখন তার কিছু কিছু টুলস সাপোর্ট এর দরকার পরবে। যেমন ফাইল সিস্টেম, অপারেটিং সিস্টেম, ইভেন্ট এরকম আরও অসংখ্য বিষয় আছে। এই জন্য নোডজেএস এর সাথে একটা ছোটোখাটো বিউল্টইন লাইব্রেরীও আছে যা আমাদের সাহায্য করে নোডজেএস ব্যবহার করে কমান্ডলাইন অ্যাপলিকেশন বা সার্ভার সাইড অ্যাপলিকেশন তৈরি করতে। যেহেতু এখানে ল্যাংগুয়েজ হিসেবে জাভাস্ক্রিপ্টই ব্যবহার করা

---

হচ্ছে তাই এই মডিউল গুলো শিখতে খুব একটা সময় লাগবে না। একে জাভাস্ক্রিপ্টের একটা নতুন লাইব্রেরী এর মত মনে করলেই হবে।

নোডজেএস বর্তমানে ওয়েব অ্যাপলিকেশন তৈরির ক্ষেত্রে খুবই জনপ্রিয়। এই জনপ্রিয়তার প্রধান কারণ এর আর্কিটেকচার। নোডজেএস সিঙ্গেল থ্রেডেড নন ব্লকিং আইও, নোডজেএস কে ডিফাইন করার জন্য সবাই এই একটা টামহি ব্যবহার করে থাকে। কিন্তু এখানে আসলে কি বোঝানো হচ্ছে? সিঙ্গেল থ্রেডেড মানে হল নোডজেএস কম্পিউটারের একটি মাত্র থ্রেড ব্যবহার করবে যখন একটা অ্যাপলিকেশন রান করা হবে। যদি আপনার কম্পিউটারে ২, ৪, ৬, ৮ টা থ্রেডও থাকে নোডজেএস বাকি একটা থ্রেডও ব্যবহার করবে না। নন ব্লকিং মানে আমরা সবাই বুঝতে পারছি যে এটা ব্লক করবে না। কিন্তু কি ব্লক করবে না? ইনপুট এবং আউটপুট ব্লক করবে না। এখানে ইনপুট আর আউটপুট হচ্ছে রিকুয়েষ্ট এবং রেসপন্স। আরও সহজ ভাবে ব্যাপারটা বোঝার চেষ্টা করা যাক।

ধরুন আপনি একটি রেস্টুরেন্টে গিয়েছেন লাক্ষ করতে। সেখানে গিয়ে দখলেন একটা মাত্র ওয়েটার আর কাস্টমার আছে পাঁচজন। ওয়েটারটাও চরম বোকা। সে একজনের কাছে যাচ্ছে, তার অর্ডার নিচ্ছে, অর্ডারটা শেফের কাছে পৌঁছে দিচ্ছে। শেফ অর্ডারটা পাওয়ার পরেও বোকা ওয়েটার ওখানেই দাঁড়িয়ে থাকছে যতক্ষণনা পর্যন্ত ওই কাস্টমারের অর্ডারটি রেডি হচ্ছে। অর্ডার রেডি হওয়ার পরে সে কাস্টমারকে অর্ডারটি বুঝিয়ে দিয়ে দ্বিতীয় কাস্টমারের কাছে যাচ্ছে। আর কাস্টমারদেরকে অপ্রয়োজনীয় ভাবে বসিয়ে রাখছে। সবাই বসে চেঁচামেচি করছে, কেউ কেউ উঠে রাগ করে চলে যাচ্ছে। এই যে কনসেপ্টটা একে বলে ব্লকিং আইও। কারণ এটা রিকুয়েষ্ট ব্লক করে। একটা রিকুয়েষ্ট আসলে সেটার রেসপন্স পাঠানোর পূর্বে সার্ভার আর কোনো নতুন রেসপন্স গ্রহণ করতে পারে না। ট্রেডিশনাল সব প্রোগ্রামিং ল্যাংগুয়েজই এই ভাবে কাজ করে থাকে। জাভা, সি++, সিশার্প, পাইথন, পিএইচপি সবাই এভাবে কাজ করে থাকে। তবে এই ল্যাংগুয়েজ গুলো মাল্টিথ্রেড হওয়ায় কিছুটা সমস্যার সমাধান করা যায়। ধরেন একজন ওয়েটারের বদলে আপনার যত গুলো টেবিল ততজন ওয়েটার যদি আপনি রাখেন তাহলে অন্ততপক্ষে কেউ বিরক্ত হয়ে চলে যাবে না। কিন্তু তারপরেও সমস্যার সৃষ্টি হবে। কারণ একটা রেস্টুরেন্টে আপনি জানেন যে আপনার কয়টা টেবিল আছে আর একসাথে সর্বোচ্চ আপনার কয়জনকে সার্ভ করতে হবে। কিন্তু ওয়েব অ্যাপলিকেশনের ক্ষেত্রে এই নাস্বারটা অজানা। যদি আপনি অনেক কম থ্রেডেড সার্ভার ব্যবহার করেন তাহলে দেখা যাচ্ছে ইউজার উপরে পরে, আবার অনেক বড়সড় সার্ভার নিলেন, কিন্তু দেখা যাচ্ছে বেশির ভাগ সময়ই সার্ভার ফাকা পরে থাকছে কিন্তু আপনাকে ঠিকই এর সার্ভিস চার্জ প্রদান করতে হচ্ছে।

---

ব্যাপারটা একটু ভিন্ন ভাবে ঘটলেও পারত। একজন ওয়েটার কিন্তু পাঁচজন কাস্টমার। ওয়েটার প্রথমে একজনের কাছে গিয়ে তার অর্ডার নিবে, তারপরে অর্ডারটা শেফের কাছে পৌঁছে দিয়ে ব্যাক চলে আসবে দ্বিতীয় কাস্টমারের কাছে। ওইদিকে শেফ প্রথম অর্ডারের কাজ চালিয়ে যেতে থাকবে। ওয়েটার দ্বিতীয় জনের কাছে থেকে অর্ডার নিয়ে আবার শেফের কাছে পৌঁছে দিয়ে তৃতীয় জনের কাছে চলে যাবে অর্ডার নেওয়ার জন্য। যদি এইভাবে অর্ডার নেই তাহলে কোনো কাস্টমারই অসম্ভুষ্ট হবে না। যখন প্রথম কাস্টমারের অর্ডার রেডি হয়ে যাবে তখন তার কাছে পৌঁছে দিবে। এভাবে যত কাস্টমারই আসুক তার অর্ডার নিতে থাকবে এবং অর্ডার প্রস্তুত হলে তার কাছে পৌঁছে দিবে। এই কনসেপ্টটাকে বলে নন ব্লকিং আইও।

নোডজেএস এই মডেলে কাজ করে থাকে। যখন কোনো ইউজার রিকুয়েষ্ট পাঠায় তার রিকুয়েষ্ট প্রসেস করার সময় নোডজেএস অন্য রিকুয়েষ্টকে ব্লক করে না। বরঞ্চ তাকে কিউইতে রেখে দেয়, এবং যখন কোনো রিকুয়েষ্ট প্রোসেস করা শেষ হয়ে যায় তখন তাকে রেসপন্স ব্যাক করে দেয়। আর এইজন্যই নোডজেএস এর থেকে ফাস্ট কোনো ব্যাকেন্ড সার্ভিস নেই। ডাটা ড্রাইভেন অ্যাপলিকেশন বানানোর জন্য নোডজেস সব থেকে ভালো সমাধান। কারণ এইক্ষেত্রে সিপিইউ এর তেমন কোনো কাজ থাকে না, সিপিইউ কয়েক মিলিসেকেন্ডের ভিতরেই একটা রেসপন্স জেনারেট করে ফেলতে পারে। এই জন্য এক সেকেন্ডে কয়েক হাজার রিকুয়েষ্ট নোডজেএস হ্যান্ডেল করতে পারে এবং খুবই দ্রুত। আপনারা খেয়াল করে দেখবেন নোডজেএস ব্যবহার করে বানানো অ্যাপলিকেশন গুলো অনেক বেশি ফাস্ট হয়।

নোডজেএস ব্যবহার করে ওয়েব অ্যাপলিকেশন তৈরি করাও খুব সহজ। কারণ HTTP সার্ভার তৈরি করার সমস্ত টুলস নোডজেএস এর সাথে বিউল্টইন ভাবেই দেওয়া থাকে। কোনো প্রকার কোন সার্ভার টুলস যেমন Xammp, Wammp ইন্সটল করার প্রয়োজন নেই। আমি যখন প্রথম নোডজেএস ব্যবহার করে সার্ভার তৈরি করেছিলাম, কোনো ভাবেই মেনে নিতে পারছিলাম না যে আমার সার্ভার তৈরি হয়ে গিয়েছে। লোকালহোস্টে আমার অ্যাপলিকেশন রানিং এবং আমার অ্যাপলিকেশন HTTP রিকুয়েষ্ট লিসেন করছে। এটা এতেটাই সহজ। মাত্র দুই তিন লাইনের কোড লিখেই একটা সার্ভার রান করা যায়।

বিগিনার অনেকের মনেই এই প্রশ্ন ঘূরপাক থায় যে পিএইচপি ব্যবহার করে আমরা যেমন ডাইনামিক অ্যাপলিকেশন তৈরি করতে পারি, নোডজেএস ব্যবহার করে কি আমরা তা পারবো? শুধুমাত্র কোর নোডজেএস লাইব্রেরী ব্যবহার করেই আপনারা সম্পূর্ণ ডাইনামিক

---

অ্যাপলিকেশন তৈরি করতে পারবেন। তবে এই ক্ষেত্রে আপনাকে প্রচুর কোড লিখতে হবে, প্রচুর লজিক নিয়ে কাজ করতে হবে। এই ব্যাপারটাকে সহজ করার জন্য আছে Express, নোডজেএসের সব থেকে জনপ্রিয় এবং বহুল ব্যবহৃত ফ্রেমওয়ার্ক।

এক্সপ্রেস হচ্ছে একটা মাইক্রোফ্রেমওয়ার্ক। মাইক্রোফ্রেমওয়ার্ক কনসেপ্ট নিয়েও অনেকের অনেক ভুল ধারণা আছে। অনেকেই মনে করেন যেহেতু এটা মাইক্রোফ্রেমওয়ার্ক তাই মনে হয় সব কিছু এখানে করা যায় না। কিন্তু ব্যাপারটা এমন না। মাইক্রোফ্রেমওয়ার্কেও আপনি সব কিছুই করতে পারবেন। তবে এখানে মিনিমালিস্ট ভাবে একটা ইমপ্লিমেন্টেশন দাঁড় করানো হয়েছে, যখন যা আপনার দরকার তা ইমপ্লিমেন্ট করার সুযোগ রেখে দেওয়া হয়েছে এবং বাকি কাজ আপনার ওপরে ছেড়ে দেওয়া হয়েছে। লারাভেল, জ্যাংগো এই ফ্রেমওয়ার্ক গুলো কমপ্লিট ফ্রেমওয়ার্ক। এখানে আগে থেকেই সব কিছু দিয়ে দেওয়া হয়েছে, যা আপনার দরকার তাও এখানে আছে, যা আপনার দরকার না তাও আছে। এরা আগে থেকেই সমস্ত সিস্টেম ডিফাইন করে রেখেছে যার ফলে আপনার নিজের পছন্দ অনুযায়ী টুলস ব্যবহার করা এখানে ঝামেলাদায়ক। অন্য দিকে এক্সপ্রেস আপনি আপনার মন মত যা খুশি তাই করতে পারেন, যেই টুলস খুশি সেই টুলস ব্যবহার করতে পারেন। সব কিছুই এখানে করতে পারবেন। মাইক্রোফ্রেমওয়ার্ক ব্যবহার করে ডেভেলপমেন্ট শেখার আর একটা মজা হচ্ছে ডেভেলপমেন্টের প্রতিটা কাজ সম্পর্কে আপনি অবগত থাকবেন। কোথায় কি হচ্ছে, কেন হচ্ছে সব কিছুই আপনার নথদর্পনে থাকবে।

এক্সপ্রেস খুবই পাওয়ারফুল একটা ফ্রেমওয়ার্ক। এটি ব্যবহার করে আপনি পিএইচপি এর মত মাল্টিপেজ অ্যাপলিকেশন তৈরি করতে পারবেন, REST API সার্ভিস তৈরি করতে পারবেন, GraphQL সার্ভার তৈরি করতে পারবেন। নোডজেএস নন ব্লকিং হওয়ায় রিয়েল টাইম অ্যাপলিকেশনের জন্য নোডজেএস সব থেকে বেশি জনপ্রিয় আর এক্সপ্রেস ব্যবহার করে আপনি রিয়েল টাইম অ্যাপলিকেশনও তৈরি করতে পারবেন খুব সহজেই। এত কিছু তৈরি করতে পারবেন শুধুমাত্র এই ছোট্ট মাইক্রোফ্রেমওয়ার্ক ব্যবহার করে। শুনে হয়ত মনে হচ্ছে এক্সপ্রেস না জানি কত কঠিন কিছু। আসলে এক্সপ্রেস খুবই সহজ একটা ফ্রেমওয়ার্ক। এটি ব্যবহার করার জন্য আপনার পূর্ব কোনো অভিজ্ঞতা থাকলেও আপনি এক্সপ্রেস ব্যবহার করে ওয়েব অ্যাপলিকেশন তৈরি করতে পারবেন। আমি নিজেও নোডজেএসের কোর লাইব্রেরী নিয়ে ঘাটাঘাটি করেছি অনেক পরে।

---

এক্সপ্রেস ব্যবহার করে কিভাবে রাউটিং করতে হয়, কিভাবে রিকুয়েষ্ট হ্যান্ডেল করতে হয়, কিভাবে পাবলিক ডিরেক্টুরি ক্রিয়েট করতে হয় আর কিভাবে সার্ভার রানিং করতে হয়, এইরকম সহজ কিছু কাজ আছে যেগুলো আপনাকে শিখতে হবে। এক্সপ্রেস নিজে খুবই ছোট এবং এর API এক পেজের ভিতরেই লিখে শেষ করা সম্ভব বর্ণনা সহ। মাল্টিপেজ অ্যাপলিকেশন বানানোর জন্য আপনার একটা টেম্পলেট ইঞ্জিন লাগবে, এক্সপ্রেস জানে কিভাবে টেম্পলেট ইঞ্জিনের সাথে কাজ করতে হয়, আপনি শুধু আপনার পছন্দমত একটা টেম্পলেট ইঞ্জিন নিয়ে এসে অ্যাপলিকেশনে বসিয়ে দেবেন। এছাড়াও সেশন, কুকিস, ফাইল আপলোড এর জন্যও এক্সপ্রেসের রয়েছে ছোট ছোট বিউলিটিন ইউটিলিটি টুলস। এক্সপ্রেসের সাথে থার্ড পার্টি যে কোনো লাইব্রেরী ব্যবহার করার জন্যও এক্সট্রা কোনো স্কিলের দরকার হয় না।

ওয়েবসাইট বা অ্যাপলিকেশন যায়ই বলেন না কেন, তৈরি করতে আপনার দরকার হবে একটা পার্সিস্টেন্ট ডাটা স্টোরেজ বা ডাটাবেজ। একটা অ্যাপলিকেশনে প্রচুর ডাটা থাকে আর সেই ডাটা রাখার জন্য রয়েছে ডাটাবেজ। এক্সপ্রেসের সাথে ডাটাবেজ হিসেবে আপনি যে কোনো জনপ্রিয় ডাটাবেজকেই বেছে নিতে পারেন। এইক্ষেত্রে কোনো ধরাবাঁধা নিয়ম নেই। ডাটাবেজের সাথে সহজে কাজ করার জন্য আছে বিভিন্ন ইউটিলিটি লাইব্রেরী যা মূলত ORM বা Object Relational Mapping নামে পরিচিত। এই টুলস গুলো ব্যবহার করে আপনি খুব সহজেই ডাটাবেজ নিয়ে কাজ করতে পারবেন। এখানে বিগিনারদের মনে একটা কনফিউশন কাজ করে থাকে, ডাটাবেজ তো শিখবো কিন্তু শিখবো কোনটা? অনেকেই মনে করে থাকেন যে PHP এর সাথে যেমন MySQL শিখতে হয়, ঠিক তেমনি NodeJS এর সাথে নিশ্চয় MongoDB শিখতে হয়। ব্যাপারটা মোটেও এরকম না।

আপনি নোডজেএস এর সাথে যে কোনো ডাটাবেজ নিয়েই কাজ করতে পারবেন। তবে মঙ্গোডিবি নিয়ে কাজ করার প্রধান কারণ হচ্ছে এটা NoSQL, মানে এখানে কোনো কুয়েরি ল্যাংগুয়েজ নেই। যা করা হয় সব কিছুই ফাংশন কলের মাধ্যমে করা হয়। আর সব থেকে বড় বিষয় এখানে সব কিছুই হয় জাভাস্ক্রিপ্ট সিনট্যাক্স ব্যবহার করে। এখানে কোনো টেবিল নেই। আছে ডকুমেন্ট যা মূলত একটা জাভাস্ক্রিপ্ট অবজেক্ট। মঙ্গোডিবিতে ডাটা স্টোর করা, কুয়েরি করা, ইন্সার্ট ডিলিট করা, যায়ই করা হোক না কেন সেটা করা হয় জাভাস্ক্রিপ্ট সিনট্যাক্স ব্যবহার করে। তাই আপনি যদি নোডজেএস এর সাথে মঙ্গোডিবি ব্যবহার করেন তাহলে এটা একটা জাভাস্ক্রিপ্ট স্ট্যাক হয়ে যায়। নতুন করে কোনো ল্যাংগুয়েজ না শিখেই সব কাজ সম্পন্ন করা যায়। আর SQL ডাটাবেজে যেই কাজ করা যায় NoSQL ডাটাবেজেও একই

---

কাজ করা যায়। NoSQL ডাটাবেজ SQL ডাটাবেজের থেকে সহজে স্কেলও করা যায়। সব মিলিয়েই মানুষ এখন মঙ্গোড়িবি কে বেশি পছন্দ করছে।

বর্তমানে ব্যাকেন্ড ডেভেলপারদের যেই কাজটা সব থেকে বেশি করতে হয় তাহল এপিআই তৈরি করা। এপিআই বলতে এখানে আমি REST API কেই বুঝাচ্ছি। এক্সপ্রেস ব্যবহার করে এপিআই তৈরি করার জন্য আপনাকে নতুন করে কোনো কিছুই শিখতে হবে না। শুধু আপনাকে ওপেন এপিআই স্পেসিফিকেশন সম্পর্কে জানতে হবে। এটা একটা স্ট্যান্ডার্ড স্পেসিফিকেশন যা মনেই সবাই এপিআই তৈরি করে থাকে। এর সাথে সাথে আপনি জেনে নিবেন কিভাবে এই স্পেসিফিকেশন মেনে এপিআই ডকুমেন্টেশন তৈরি করা যায়। বাজারে অসংখ্য টুলস আছে যা ব্যবহার করে অটোমেটিক ভাবেই ডকুমেন্টেশন তৈরি করা যায়। তার ভিতরে জনপ্রিয় একটা সমাধান হচ্ছে Swagger. এর সাথে সাথে আপনাকে শিখতে হবে কিভাবে আপনি এপিআই টেস্টিং করতে পারেন। এপিআই টেস্ট করার জন্য আপনি ব্যবহার করতে পারেন Insomnia বা Postman এর মত টুলস গুলো।

রেষ্ট এপিআই নিয়ে কিছুটা সময় ব্যয় করা উচিত সব রকম এপিআই তৈরি করার একটা অভিজ্ঞতা অর্জনের জন্য। এই সময় আপনাকে শিখতে হবে কিভাবে আপনি এপিআই অথেন্টিকেশন সার্ভিস প্রোভাইড করতে পারেন JWT ব্যবহার করে। এর সাথে সাথে বিভিন্ন অথেন্টিকেশন টুলস যেমন PassportJS কিভাবে ব্যবহার করতে হয়, কিভাবে মাল্টি অথেন্টিকেশন ইমপ্লিমেন্ট করতে হয় OAuth 2 ব্যবহার করে। কিভাবে ফাইল আপলোড করতে হয়, কিভাবে স্ট্রিম রেসপন্স পাঠানো যায়, এগুলোও শিখে নিতে হবে।

রেষ্ট এপিআই নিয়ে ভালো একটা জ্ঞান অর্জন হয়ে গেলেই কিছু থার্ড পার্টি ইন্টিগ্রেশন শিখে নিতে হবে। সবার প্রথমে শিখতে পারেন Socket.io যা ব্যবহার করে রিয়েল টাইম ফিচার যুক্ত করতে পারবেন আপনার অ্যাপলিকেশনে। যেমন রিয়েল টাইম ম্যাসেজিং, রিয়েল টাইম পুশ নোটিফিকেশন। তবে এই ধরনের সার্ভিস যে সব সময় আপনাকে ম্যানুয়ালি তৈরি করতে হবে এমন না। প্রচুর সার্ভিস প্রোভাইডার এই ধরনের সার্ভিস সেল করে থাকে যা শুধুমাত্র ইন্টিগ্রেট করতে পারলেই আপনি অনেক বড় বড় অ্যাপলিকেশনও ডেভেলপ করতে পারবেন। আমার মতে প্রথমেই কোনো সার্ভিস প্রোভাইডারের কাছে যাওয়ার দরকার নেই, ছোট পরিসরে হলেও শেখার জন্য সব কিছুই একবার করে নিজে ইমপ্লিমেন্ট করে রাখা ভালো। পরে প্রফেশনাল কাজের ক্ষেত্রে, যেখানে পার্ফরমেন্স এবং স্কেলিং একটা বড় বিষয়, সেখানে ভালো কোনো সার্ভিস প্রোভাইডারের সার্ভিস ব্যবহার করা যাতে পারে।

---

এরপরে আপনি শিখতে পারেন কিভাবে ইমেইল নিয়ে কাজ করতে হয়, কিভাবে বিভিন্ন SMTP সার্ভার আপনার অ্যাপ্লিকেশনের সাথে যুক্ত করতে হয়, কিভাবে বাল্ক ইমেইল সেন্ড করতে হয়। ইমেইলের সাথে সাথে SMS নিয়ে কাজ করাও আপনাকে শিখতে হবে। কারণ এখন মানুষ ইমেইলের থেকে SMS কেই বেশি প্রাধান্য দিয়ে থাকে। SMS Alert, Two Factor Authentication, Verification Code এই টাইপের কাজ গুলো কিভাবে করা যেতে পারে সেই বিষয়ে একটা দক্ষতা অর্জন করে নেওয়া ভালো। এছাড়াও বিভিন্ন থার্ড পার্টি স্টোরেজ সার্ভিস, সেন্ট্রাল লগ সার্ভিস, ক্যাশিং এর মত বিষয় গুলো সম্পর্কেও জ্ঞান অর্জন করতে হবে এবং সেগুলো নিজে থেকে ছোট পরিসরে ইমপ্লিমেন্ট করার অভিজ্ঞতা অর্জন করতে হবে।

বর্তমানে রেস্ট এপিআই এর সাথে পাল্লা দিয়ে আর একটা টেকনলজি ট্রেন্ডিং এ যাচ্ছে আর তা হল GraphQL যা মূলত একটা কুয়েরি ল্যাংগুয়েজ এবং এপিআই কুয়েরি করার জন্য কাজে ব্যবহার করা হয়। রেস্ট এপিআই এর ক্ষেত্রে যেমন প্রতিটা রিসোর্সের জন্য নির্দিষ্ট একটা URI থাকে, এখানে সম্পূর্ণ রিসোর্সেস জন্য একটা মাত্র URI আছে যা ব্যবহার করেই আপনি যে কোনো ডাটা তুলে আনতে পারবেন। এর আর একটা এডভান্টেজ হচ্ছে যতটুকু ডাটা আপনার দরকার ঠিক ততটুকু ডাটাই আপনি কুয়েরি করে আনতে পারবেন। একটু বেশি ও না, একটু কমও না। এই সমস্ত কিছু এডভান্টেজের কারণেই GraphQL দিন দিন জনপ্রিয় হয়ে উঠছে। আর GraphQL ব্যবহার করার কোনো লিমিটেশন নেই, আপনি যেকোনো ল্যাংগুয়েজের সাথেই এটা ব্যবহার করতে পারবেন। তাই আমার মনে হয় GraphQL শিখে নিলে এই জ্ঞানটা আপনার বিফলে যাবে না, বরঞ্চ এডভান্টেজই পাবেন সব জায়গায়।

ব্যাকেন্ড হচ্ছে আপনার অ্যাপ্লিকেশনের সব কিছু। তাই এখানে কি করা লাগবে, কি শেখা লাগবে সেটা আগে থেকেই বলে দেওয়া যায় না। ওপরে যেই বিষয় গুলো নিয়ে আলোচনা করা হয়েছে সেগুলো খুবই কমন বিষয় ব্যাকেন্ডের জন্য। যেকোনো অ্যাপ্লিকেশনের জন্যই এই বিষয় গুলো আপনার দরকার হবে। কিন্তু শুধু মাত্র এগুলো শিখে বসে থাকলেই চলবে না, ইন্টারেক্শনের প্রয়োজন হতে পারে। আপনার ইন্টার্ফেস যদি একটা ভিডিও স্ট্রিমিং সাইট চাই, সেখানে আপনাকে ভিন্ন ভিন্ন ভিডিও ফর্মেট নিয়ে কাজ করা লাগতে পারে। ইন্টার্ফেস যদি বলে ইনভয়েস এর পিডিএফ ইমেইল করতে হবে সেক্ষেত্রে পিডিএফ জেনারেশন নিয়ে কাজ করা লাগতে পারে। ইন্টার্ফেস যদি বলে বাল্ক প্রোডাক্ট ইম্পোর্ট করতে হবে সেক্ষেত্রে CSV ফাইল নিয়ে কাজ করা লাগতে পারে। ইউজার গুগল, ফেসবুক, অ্যামাজন এবং

---

সার্ভিস ব্যবহার করতে বলতে পারে। ওয়েব স্ক্রাপিং করার প্রয়োজন হতে পারে কোনো অ্যাপ্লিকেশনের জন্য। ব্যাকেন্ডের সম্ভাবনা আসলে বলে শেষ করার মত না। আপনার নিজের মানুষিকতাকে এভাবেই তৈরি করতে হবে যেন আপনি যখন যা লাগে তা শিখে নিতে পারেন। ওপরে বর্ণিত বিষয় গুলো হল আপনার ব্যাকেন্ডের বেস। এই বেস শুধু নোডজেএস এর জন্য না। যেকোনো ল্যাংগুয়েজ বা ফ্রেমওয়ার্কেই আপনার এই জ্ঞান কাজে লাগবে। শুধুমাত্র ল্যাংগুয়েজ এবং ইমপ্লিমেন্টেশনের ধারাটা পরিবর্তিত হয়ে যাবে।

ব্যাকেন্ডের সাথে আর একটা বিষয়ও এখন গুরুত্বপূর্ণ। আর সেটা হচ্ছে DevOps। নতুন একটা টার্মস যার মানে হচ্ছে ডেভেলপমেন্ট অপারেশনস। এটা সম্পূর্ণ নতুন একটা টাইটেল, যারা কাজ করে ডেভেলমেন্ট এবং প্রোডাকশনের মাঝখানে। যাদের কাজ একটা অ্যাপ্লিকেশন সঠিকভাবে ডেভেলপড হওয়ার পরে আটোমেটেড ভাবে টেষ্টিং, স্টেজিং করে প্রোডাকশনে ডেপলয় করা। একজন ডেভেলপার হিসেবে DevOps সম্পর্কে অল্প কিছু জ্ঞান আমাদের রাখা উচিত। যেমন আটোমেটেড টেষ্টিং। একটা কোড লেখার পরে তার ইউনিট টেষ্ট, ইন্টিগ্রেশন টেষ্ট করাটা একজন ডেভেলপারের জন্য জরুরি। এর জন্য আপনার বিভিন্ন আটোমেটেড টেষ্ট টুলস সম্পর্কে ধারণা রাখতে হবে। এর সাথে সাথে CI/CD টুলস যেমন TravisCI, CircleCI সম্পর্কেও একটা ধারণা রাখা উচিত।

বর্তমানে আর একটা টেকনোলজিও খুব জনপ্রিয় হয়েছে আর সেটা হচ্ছে Docker। ভার্চ্যাল বক্স এর জামানা শেষ করে এখন সবাই দৌড়াচ্ছে কন্টেইনারের পিছনে। ডকার খুবই সুন্দর এবং জনপ্রিয় একটা টেকনোলজি হওয়ায় এখন সবাই প্রায় তাদের অ্যাপ্লিকেশন ডকার এবং কুবারনেট ব্যবহার করেই ডেপলয় করছে। ডকার হচ্ছে একটা আইসোলেটেড এনভাইরনমেন্ট যার ভিতরে আপনার অ্যাপ্লিকশন সহ এটি রান করার সমস্ত ডিপেন্ডেন্সি আইসোলেটেড ভাবে থাকে। কথা গুলো আসলে একটু জটিল বিগিনারদের জন্য। তবে এটা প্রথমেই আপনাকে শিখতে হবে না, বা একবারেই শিখে বস হয়ে যেতে হবে না। একজন ডেভেলপার হিসেবে ডকার সম্পর্কে নৃন্যতম একটা জ্ঞান রাখা আমাদের সবার জন্যই জরুরি।

শুধু এতেটুকুই না, আরও অনেক কিছু আছে। DevOps, Server, System Architecture, Infrastructure আরও কত কি? এই বিষয় গুলো শিখতে বছরের পর বছর সময় লেগে যায়। এত দিন পর্যন্ত আমরা যা শিখলাম তা আপনাকে একজন ভালো ডেভেলপার বানাবে। আপনি এই জ্ঞান কাজে লাগিয়েই যেকোনো অ্যাপ্লিকেশন তৈরি করতে পারবেন। কিন্তু এই অ্যাপ্লিকেশনটা যখন কোটি কোটি মানুষের কাছে পৌঁছে দিতে চাহিবেন তখন দরকার হবে

সিস্টেম আর্কিটেকচারের মত বিষয়। কিন্তু এখনই আপনার এই বিষয় গুলো নিয়ে চিন্তা করার কোনো দরকার নেই। আপনি যদি প্রথম থেকে সমস্ত গাইড লাইন মেনে এই পর্যন্ত আসেন, সেটা যেই কয় বছরেই আসেন না কেন, এই পর্যন্ত আসার পরে আপনার আর কোনো গাইডলাইনের দরকার হবে না। তখন প্রয়োজনের তাগিদে আপনি নিজেই সব কিছু শিখে নিতে পারবেন।

ফুলস্ট্যাক ডেভেলপার হওয়ার জানিটা একটা বিরাট জার্নি। এই জার্নি আপনি অল্প কয়েকদিনেও শেষ করে ফেলতে পারেন। অল্প কয়েক দিনেই আপনি নিজেকে ফুলস্ট্যাক ডেভেলপার বলে দাবি করতেই পারেন। তবে যেকোনো অ্যাপ্লিকেশন ডেভেলপ করার মত যোগ্যতা তৈরি হতে অনেক সময় লাগে। একটা ব্লগ সাইটের ফ্রন্টেন্ড, ব্যাকেন্ড নিজে নিজে তৈরি করে যেকোনো একটা Pass সার্ভিসে ডেপলয় করলে দুনিয়াতো আপনাকে একজন ফুলস্ট্যাক ডেভেলপার বলবেই। আপনি আসলেই ফুলস্ট্যাক ডেভেলপার, কারণ আপনি সম্পূর্ণ প্রোসেসটা জানেন এবং করতে পারেন। কিন্তু শুধুমাত্র আপনিই জানেন আপনি কতটা নিজেকে তৈরি করতে পেরেছেন। এইজগতে শিক্ষার কোনো বিকল্প নেই, প্রতিটা রাতই পার করতে হয় নতুন কিছু না কিছু শিখতে, তবে এই শিক্ষার পিছনে আছে ভালোবাসা, আছে নেশা।

## NodeJS এ যা যা শিখতে হবে

- NodeJS Architecture
- NodeJS Builtin Modules
- Express Frameworks
- Route and Middleware
- Template Engines
- Session & Cookies
- Authentication Techniques
- File Upload and Public Directory
- Error Handling & Debugging
- REST API Specifications
- MongoDB with Mongoose
- MySQL with Sequelize
- Email with Nodemailer
- Twilio, Sendgrid and Pusher
- Socket.io
- GraphQL, Prisma & Apollo
- Cloudinary, AWS S3, Vimeo
- Test Driven Development
- NodeJS Design Patterns

# NodeJS শিখার রেফারেন্স

## Must Read NodeJS:

- NodeJS in Action By Mike Cantelon
- Express in Action By Evan Hahn
- Professional NodeJS: Building Javascript Based Scalable Software By Pedro
- Practical NodeJS: Building Real World Scalable Web Apps By Azat Mardan
- Node: Up & Running By Mike Wilson
- Node.js Design Patterns By Mario Casciaro

## Youtube Channels to NodeJS

- [Web Developer BD](#) (Bangla)
- [Stack Learner](#) (Bangla)
- [Academind](#) (English)
- [Programming Knowledge](#) (English)
- [The Net Ninja](#) (English)
- [Codedamn](#) (English)

## Websites to Learn NodeJS

- [NodeJS Official Docs](#)
- [NodeJS Dev](#)
- [Node School](#)

## Training Program for NodeJS

- [Stack Learner Premium Courses](#)
- [Stack Learner Offline Bootcamps](#)



এই পেজটি স্বেচ্ছায় ফাঁকা রাখা হয়েছে

Visit	<a href="https://courses.stackschool.co">https://courses.stackschool.co</a>
Subscribe	<a href="https://youtube.com/stacklearner">https://youtube.com/stacklearner</a>
Like	<a href="https://facebook.com/stacklearner">https://facebook.com/stacklearner</a>
Join	<a href="https://facebook.com/groups/stacklearner">https://facebook.com/groups/stacklearner</a>
Connect	<a href="https://linkedin.com/company/stacklearner">https://linkedin.com/company/stacklearner</a>
Follow	<a href="https://instagram.com/stacklearner/">https://instagram.com/stacklearner/</a>
Follow	<a href="https://medium.com/stack-learner">https://medium.com/stack-learner</a>

১০

---

অধ্যায় দশ

শেষ কথা

## এই অধ্যায়ে আলোচিত বিষয়বস্তু

- ✓ ভুল ধারণা
- ✓ আমাদের কি করা উচিত
- ✓ কেন আমাদের মডার্ন টেকনোলজি শেখা উচিত

---

স্বাগতম, ডেভেলপমেন্টের এই বিভীষিকাময় যাত্রা পার করে আপনি এই পর্যন্ত এসেছেন এই জন্য আপনাকে স্বাগতম। এই পর্যন্ত আসার মাঝে আপনি অনেক টেকনোলজি এর নাম শুনেছেন, অনেক ল্যাংগুয়েজ, অনেক ফ্রেমওয়ার্কের নাম শুনেছেন। অনেকের মনে হতে পারে আমি তব দেখানোর জন্যই এই সব লিখে রেখেছি, অনেক টেকনোলজিকে অসম্মান করেছি। কিন্তু আমি কোনোটাই করি নি। আপনাদের মনে যত সংশয় ঘূরপাক থাচ্ছে, সমস্ত সংশয় সমস্ত ভুল আমি ভাঙ্গার চেষ্টা করবো এখানে।

ডেভেলপমেন্ট একটা বৃহৎ যাত্রা। কয়েক বছর সময় লাগে এই যাত্রা সম্পন্ন করে নিজেকে ভালো একজন ডেভেলপারের কাতারে নিয়ে যেতে। এত এত টেকনোলজি, এত এত নাম সব আপনাকে একদিনে শিখতে হবে না। আপনি প্রথমে ভালো ভাবে প্রোগ্রামিং ল্যাংগুয়েজ শিখবেন, ডাটা স্ট্রাকচার অ্যালগোরিদম শিখবেন, তারপরেই ডেভেলপমেন্ট আসবেন। প্রোগ্রামিং শেখার পিছনে, প্রৱেশ সন্তুষ্টি করার পিছনেই আপনার সব থেকে বেশি সময় বয় হবে। ব্যাপারটা একটা সহজ উদাহরণের মধ্য দিয়ে বোঝার চেষ্টা করা যাক। আপনাকে যদি একটা ভেঁতা কুড়াল দিয়ে বড় একটা গাছ কাটতে বলা হয় আপনি পারবেন? হ্যত পারবেন কিন্তু প্রচুর কষ্ট করতে হবে। তার থেকে বরঞ্চ সরাসরি গাছ কাটার কাজ শুরু না করে, প্রথমে সময় নিয়ে যদি কুড়ালটা ধার করা হয় এবং ধারালো কুড়াল দিয়ে গাছটা কাটার চেষ্টা করা হয় তাহলে আপনি খুব দ্রুত গাছটি কাটতে পারবেন। [নোটঃ গাছ পরিবেশের বন্ধু, অপ্রয়োজনে গাছ কাটা থেকে বিরত থাকুন]

প্রোগ্রামিং আর ডেভেলপমেন্টও একই রকম। আপনি যত ভালো প্রোগ্রামার হবেন, তত ভালো ডেভেলপমেন্ট করতে পারবেন। তাই সময় লাগলেও প্রোগ্রামিং এবং প্রৱেশ সন্তুষ্টি এর জ্ঞানটা সবার প্রথমেই অর্জন করে নেওয়া উচিত। ডেভেলপমেন্টে আপনি প্রচুর টুলস পাবেন, যা আপনি ধীরে ধীরে শিখবেন। এমনটা না যে সব গুলো না শিখলে কাজ করা যাবে না। আপনি অল্প শিখবেন, কাজ করবেন আবার নিজেকে আপগ্রেড করবেন। একটা সুন্দর প্রোসেসের ভিতর দিয়ে আগাতে হবে আপনাকে। তাহলে কাজও হবে, উপার্জনও হবে, নতুন নতুন বিষয় শেখাও হবে। প্রতিদিন ১ ঘণ্টা বা ২ ঘণ্টা সময় বরাদ্দ থাকবে শুধু মাত্র নতুন কিছু শেখার জন্য।

ডেভেলপমেন্টের এই যাত্রায় হাজার হাজার টেকনোলজি আমাদের সামনে আসবে। কোনো টেকনোলজি কারোর কম্পিউটর না। কেউ ছেট বা বড় না, কোনোটা জটিল বা সহজ না। আমরা যতটা পারবো ততটাই শেখার চেষ্টা করবো। কাবণ সমস্ত টেকনোলজির, সমস্ত টুলস

---

এর একটা মূল্য আছে। বাজারে এর চাহিদা আছে, ব্যবহার আছে। এর ভিতরে একটা যদি আমরা না জানি তাহলে ওই সেক্টরটা আমরা মিস করে যাবো। আমি একবারেই সব কিছু শিখতে বলছি না। প্রথমেই আপনাকে যেকোনো একটা ভালো স্ট্যাক বেছে নিতে হবে। সেই স্ট্যাক এর জনপ্রিয় সমস্ত টুলসই এক এক করে আপনাকে এক্সপ্লোর করতে হবে। নিজের সামর্থ্য অনুযায়ী শিখতে থাকতে হবে। তাহলে দুইটা জিনিস হবে, প্রথমত আপনার অভিজ্ঞতা বৃদ্ধি পাবে আর দ্বিতীয়ত আপনি আপনার কম্পিউটারের থেকে কিছুটা এগিয়ে থাকবেন। তবে নতুন করে যেই বিষয়টায় শেখা শুরু করেন না কেন, সেটা ভালো ভাবেই শিখতে হবে। যেমন তেমন করে শিখলে কোনো লাভ নেই।

টেকনোলজি স্ট্যাক বেছে নেওয়ার সময়ে আপনাকে যথেষ্ট সাবধান থাকতে হবে। আপনি যেই স্ট্যাক এ কাজ করবেন বলে সিদ্ধান্ত নিচ্ছন তার ডিমান্ড কেমন, সেখানে উপর্যুক্ত কেমন, কত মানুষ সেখানে কাজ করছে, কাজের তুলনায় কাজ করার মানুষের অনুপাত কেমন এই বিষয় গুলো মাথায় রেখেই আপনাকে সিদ্ধান্ত নিতে হবে। কাজ করার ক্ষেত্রে কোনো টেকনোলজি সহজ বা কঠিন হয় না, সবই এক। আপনারা অনেকেই হ্যাত ভাবছেন যে আমি পিএইচপি ওয়ার্ডপ্রেস পছল করি না। সব সময় এর বিরুদ্ধে কথা বলি। আসলে আমি কোনো টেকনোলজি এর বিপক্ষে না। আমি নিজেও প্রযোজনের তাগিদে পিএইচপি বা ওয়ার্ডপ্রেস ব্যবহার করে থাকি। আমার পয়েন্টটা ভিন্ন জায়গাতে। একটা জাতির সবাই এই এক টেকনোলজি নিয়ে কাজ করবে? তাহলে সফটওয়্যার ইন্ডাস্ট্রি গড়ে উঠবে কেমন করে? পিএইচপি ওয়ার্ডপ্রেস দিয়ে আর যায় হোক সফটওয়্যার ইন্ডাস্ট্রি গড়ে তোলা যায় না। তাই আমি সবাইকে এক দিকে যেতে না করেছি। কিছু মানুষ তো থাকে যাদের হাজারে না, কোটিতে স্বপ্ন।

আমি ফ্রিলান্সিং নিয়েও অনেক কথা বলেছি। আমি তাদেরকে উদ্দেশ্য করে কিছুই বলিনি যারা আমাদের দেশের গর্ব, প্রতি মাসে হাজার হাজার বৈদেশিক মুদ্রা দেশে নিয়ে আসছে। আপনি ফ্রিলান্সিং করে নিজের কর্মসংস্থান নিজে করলে বরঞ্চ আমাদের দেশের জন্যই ভালো। কিন্তু জ্ঞান বৃদ্ধি অর্জন না করে, নিজেকে দক্ষ না করে যারা ফ্রিলান্সিং করতে যাচ্ছে তারা তো দক্ষদের জন্যও ভুমকি সরূপ হয়ে দাঁড়াচ্ছে। ফ্রিলান্সিং করার জন্য আপনাকে যে কোনো একটা বিষয়ে ভালো দক্ষতা অর্জন করতে হবে। আর মজার বিষয় কি জানেন? এই দক্ষতা অর্জন হয়ে গেলে আপনাকে আর বিভিন্ন জায়গায় ফ্রিলান্সিং এর কোর্স করে বেড়াতে হবে না। আপনি নিজে নিজেই ফ্রিলান্সিং এর পুরো ব্যাপারটা উদ্ঘাটন করতে পারবেন। তাই ফ্রিলান্সিং এর পিছনে না ছুটে, দক্ষতার পিছনে দৌড়ান।

---

আমাদের দেশে এখন অস্ত্রকাঁধে দাঁড়িয়ে থাকা বলিষ্ঠ সৈনিকের থেকে বেশি দরকার কম্পিউটারের সামনে বসে কীবোর্ডের বাটনে সুর তুলতে পারা সৈনিক। অন্য আর দশটা সেক্টরে না পারলেও আমরা কিন্তু পারি সফটওয়্যার সেক্টরে সারা পৃথিবীকে তাক লাগিয়ে দিতে। আমরা করছিও অনেকাংশে। পৃথিবীর বহু দেশের সফটওয়্যার আমাদের দেশে তৈরি হচ্ছে। ভবিষ্যতে আরও হবে। আর এর জন্য আমাকে, আপনাকে সবাইকে এগিয়ে আসতে হবে। একটা কথা আপনাকে মনে রাখতে হবে, দেশের জন্য যুদ্ধ করাটাই শুধু দেশপ্রেম না। দেশের একজন সচেতন নাগরিক হয়ে বেঁচে থাকাটাও বড় রকমের দেশপ্রেম। আপনার দেশকে ভালোবাসার জন্য দেশের জন্য যুদ্ধ করার দরকার নেই। শুধুমাত্র নিজের জন্য যুদ্ধ করুন। আপনি যদি নিজেকে স্বাবলম্বী করতে পারেন, নিজেকে অর্থনৈতিক ভাবে স্বচ্ছল করতে পারেন তাহলে এটাই হবে আপনার দেওয়া দেশের জন্য সব থেকে বড় উপাহার।

## ধন্যবাদ

## লেখকের কথা

আমি হাসান মাহমুদ নায়েম (HM Nayem), খুবই ক্ষুদ্রমাপের একজন ডেভেলপার। প্রোগ্রামিং, ডেভেলপমেন্ট সম্পর্কে আমার জ্ঞান খুবই সীমিত। এই বই এর মাধ্যমে আমি আমার এই ক্ষুদ্র জ্ঞান কিছু মানুষের কাছে পৌঁছে দেওয়ার চেষ্টা করেছি। নয় বছর এই জগতে যুদ্ধ করার দরুণ চোখের সামনে যেই ভুল এবং গুজব গুলো এসেছে তা নিজের মত করে ব্যাখ্যা দেওয়ার চেষ্টা করেছি। কিভাবে একজন মানুষ নিজেকে প্রস্তুত করলে ডেভেলপমেন্টের যেকোনো সেক্টরে ভালো করতে পারবে বলে আমি মনে করি সেই বিষয় গুলোই উপস্থাপন করার চেষ্টা করেছি।

আমি একজন স্বপ্নবাজ মানুষ। আমি স্বপ্ন দেখি আমাদের দেশের সফটওয়্যার ইন্ডাস্ট্রি একদিন অনেক বড় হবে। আমাদের দেশের তরুণ সমাজের তৈরি করা গেম সফটওয়্যারের চাহিদা তৈরি হবে বিশ্ব বাজারে। গুগল, ফেসবুক, অ্যামাজন, নেটফ্লিক্সের থেকে বড় বড় সার্ভিস তৈরি করবে আমাদের তরুণরা। নিজের বুদ্ধিমত্তা দিয়ে নাসাতে জব করবে না, নাসার মত বড় বড় সব আবিষ্কার করবে। দেশতো এগিয়ে যাচ্ছেই, সামনে আরও এগিয়ে যাবে। কিন্তু আমরা কি আগাতে পারছি?

আমি আজকেই সব কিছু করার কথা বলছি না। আমি বলছি পরবর্তী ২০ বছরের কথা। পরবর্তী জ্ঞানারেশন যখন দায়িত্ব নিবে, তখন যেন তাদের গোলাম হয়ে থাকতে না হয়। তারা যেন রাজা হয়ে থাকতে পারে তার প্রস্তুতি আজকেই নিতে হবে। আগামী বিশ্ব বছর পরে বিশ্ব অর্থনীতির একটা বড় অংশ যদি আমরা হতে চাই, তাহলে আজকে থেকেই আমাদের কাজ শুরু করতে হবে। একটা দেশের অবস্থান বিশ্ব অর্থনীতিতে কি হবে, তার পুরোপুরিই নির্ভর করে সেই দেশের নাগরিকের ওপরে। আর নাগরিক হিসেবে নিজের দেশকে বাঁচানোর জন্য, নিজের দেশের অর্থনৈতিক উন্নতির জন্য আমরা সব কিছু করতে প্রস্তুত। এটাই আমাদের দেশপ্রেম। অস্ত্র হাতে যুদ্ধের ময়দানে না দাঁড়াতে পারলেও, একজন সফটওয়্যার সৈনিক হিসেবে ল্যাপটপ হাতে যুদ্ধ করতে আমি সদা প্রস্তুত।