بسم الله الرحمن الرحیم

تکنولوژی کامپیوتر

جلسه اول - معرفی درس

معرفی خودم

- Alireza Tofighi
- Ph.D Student at Sharif, CTO at Balad.ir
- Email: alirtofighim+ct1403@gmail.com
- Telegram: t.me/ATofighi
- https://alireza.dev/teaching/ct1403/
- **مشاوره و رفع اشکال؟**

- به من ایمیل بزنید، تلگرام پیام دهید یا ممکن است در اتاق ۱۱۴ دانشکده یا لابی پیدایم کنید!

هدف درس

تکنولوژی کامپیوتر؟؟؟

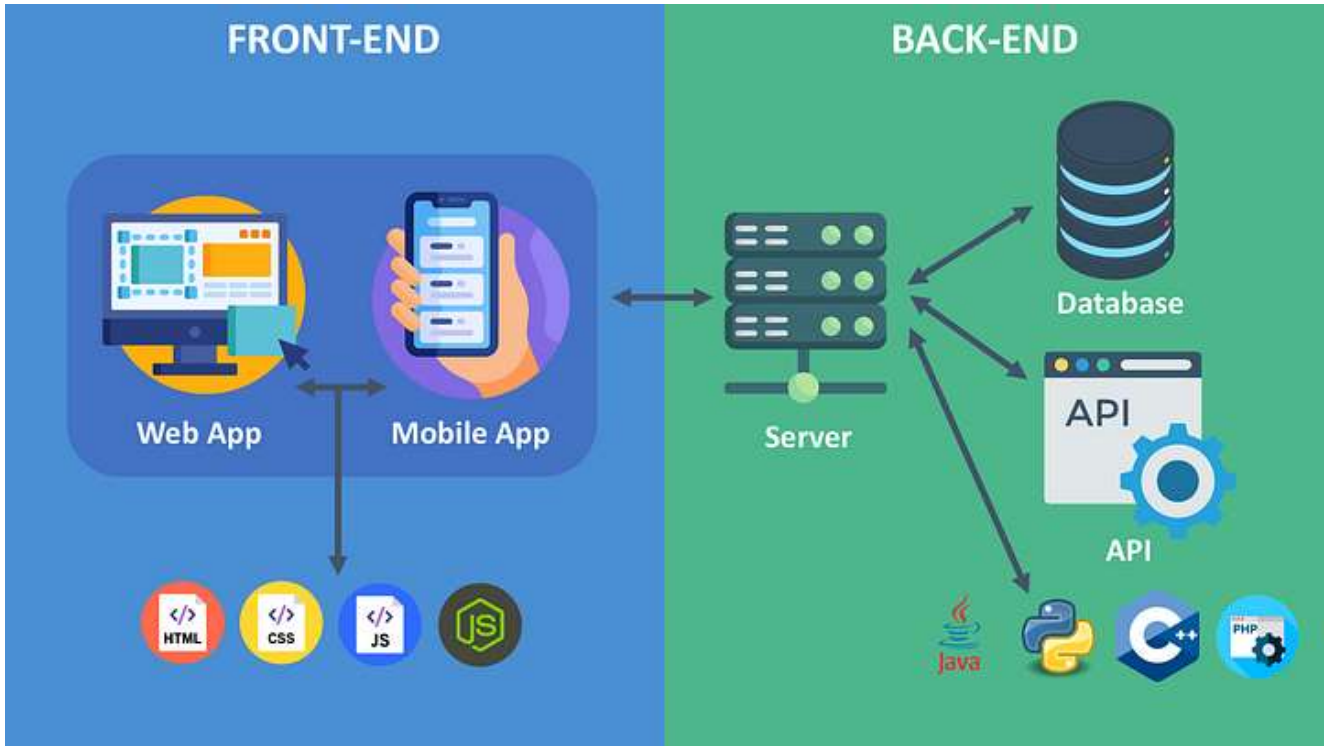
- در گذشته انواع مطالب در این درس گفته شده
  - *در یک ارائه وب*
  - *در یک ارائه IOS*
  - *در یک ارائه Big Data Engineering*
  - *...*

خب، پس تکنولوژی کامپیوتر؟؟؟؟

- این ارائه:
  - *برنامه‌نویسی وب، با تمرکز به برنامه‌نویسی سمت سرور*
  - *طراحی نرم‌افزارهای داده‌محور با داده‌های زیاد*

فرقش با برنامه‌نویسی وب؟

- فرانت‌اند در مقابل بک‌اند



فرقش با برنامه‌نویسی وب؟

- عموما تمرکز یک درس برنامه‌نویسی وب روی هر دو جنبه‌ی فرانت‌اند و بک‌اند، و بیشتر بر روی برنامه‌نویسی font-end یک سایت یا webapp خواهد بود.
- ولی ما تمرکز بیشتری روی برنامه‌نویسی backend داریم و خیلی مروری frontend را مرور می‌کنیم.

فرقش با برنامه‌نویسی وب؟

- در بک‌اند
  - *در برنامه‌نویسی وب عموما یک framework خاص، مثلا django، laravel، یا مثلا Spring انتخاب می‌شود.*
  - *ما بر روی زبان golang تمرکز می‌کنیم و مسائل کلی را یاد می‌گیریم که در انواع سیستم‌ها قابل پیاده کردن است.*

فرقش با برنامه‌نویسی وب؟

- در فرانت‌اند
  - *عموما یک framework مثل ReactJS انتخاب می‌شود و عمیق آن را بررسی می‌کنند.*
  - *ولی ما به شکل مروری روی آن رد می‌شویم.*

فرقش با مهندسی داده‌های حجیم

- در درس مهندسی داده‌های حجیم
  - *در نهایت یک مهندس داده که بلد است با داده‌های حجیم کار کند یا ابزارهای آن‌ها را بالا بیاورد.*
  - *روی داده‌ی حجیم پرسمان اجرا کند.*
- ما
  - *تمرکز بر روی روش ساختن ابزارهایی که در مقایس بالا کار کنند.*

مطالب درس

بخش اول- برنامه‌نویسی وب

جلسه	تاریخ	موضوع جلسه	تمرین	مط
۱	۱۴۰۳/۱۱/۲۰	معرفی درس، شروع زبان گولنگ		
۲	۱۴۰۳/۱۱/۲۷	زبان گولنگ		
۳	۱۴۰۳/۱۱/۲۹	زبان گولنگ		
۴	۱۴۰۳/۱۲/۰۴	شبکه	تمرین ۱: گولنگ، شبکه	
۵	۱۴۰۳/۱۲/۰۶	گولنگ و http server		
۶	۱۴۰۳/۱۲/۱۱	مقدماتی بر html و css		
۷	۱۴۰۳/۱۲/۱۳	مقدماتی بر javascript		
۸	۱۴۰۳/۱۲/۱۸	نوشتن API و REST و ...		
۹	۱۴۰۳/۱۲/۲۰	دیپلویمنت، داکر، nginx		
۱۰	۱۴۰۳/۱۲/۲۵	پایگاه داده		
۱۱	۱۴۰۳/۱۲/۲۷	پایگاه داده ۲	تمرین ۲: پیاده‌سازی یک سایت کامل با همه‌چی	
		حل تمرین - دیدن کل پیاده‌سازی‌های سایت و چند مسئله‌ی تکمیلی		
		نوروز مبارک ^_^		

بخش دوم – طراحی برنامه‌های داده‌محور

۱۲	۱۴۰۴/۰۱/۱۶	افزایش لود، مسائل مختلف، برنامه‌های قابل اتکا، نگهداری و مقیاس پذیر	
۱۳	۱۴۰۴/۰۱/۱۸	رپلیکیشن	
۱۴	۱۴۰۴/۰۱/۲۳	رپلیکیشن	
۱۵	۱۴۰۴/۰۱/۲۵	پارتیشنینگ	
	نیمه دوم فروردین	آزمون میان‌ترم!	
۱۶	۱۴۰۴/۰۱/۳۰	ترنزکشن	تمرین ۳: کلاستر کردن یک دیتابیس / پیاده‌سازی برخی از الگوریتم‌های سیستم‌های توزیع‌شده
۱۷	۱۴۰۴/۰۲/۰۱	مسائل دیگر سیستم‌های توزیع‌شده، fault، مسئله‌ی clock	
۱۸	۱۴۰۴/۰۲/۰۶	کانسیستنسی	
۱۹	۱۴۰۴/۰۲/۰۸	اجماع	
۲۰	۱۴۰۴/۰۲/۱۳	قضیه‌ی CAP و جمع‌بندی مسائل	

بخش دوم – طراحی برنامه‌های داده‌محور

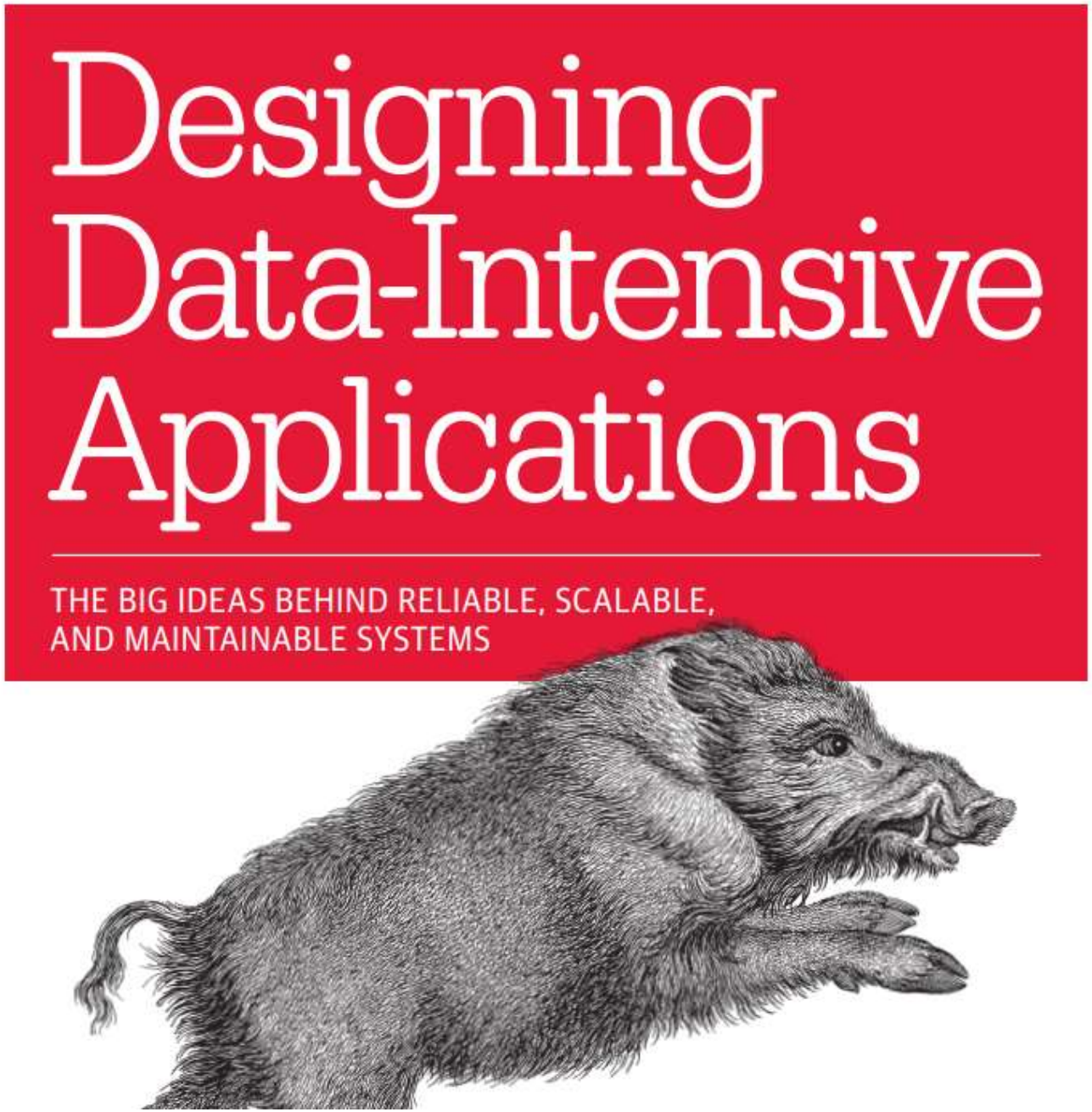
۲۱	۱۴۰۴/۰۲/۱۵	بررسی برخی از دیتابیس‌ها و نکات پیاده‌سازی	
۲۲	۱۴۰۴/۰۲/۲۰	بررسی برخی از دیتابیس‌ها و نکات پیاده‌سازی ۲	
۲۳	۱۴۰۴/۰۲/۲۲	بررسی برخی از دیتابیس‌ها و نکات پیاده‌سازی۳	تمرین ۴: بالا آوردن و استفاده از این دیتابیس‌ها با پیاده‌سازی‌های عملی
۲۴	۱۴۰۴/۰۲/۲۷	مسیج پسینگ، معماری lambda و Kafka	
۲۵	۱۴۰۴/۰۲/۲۹	Hadoop / Map Reduce	
۲۶	۱۴۰۴/۰۳/۰۳	MapReduce / Spark	
۲۷	۱۴۰۴/۰۳/۰۵	Spark	
۲۸	۱۴۰۴/۰۳/۱۰	Spark Streaming	تمرین ۵: کار کردن با ابزارهای Big Dataای، کارهای آنالیتیکی با اینا
۲۹	۱۴۰۴/۰۳/۱۲	جلسه‌ی آخر	

منبع درس

بخش اول

- متاسفانه از روی درس‌نامه یا کتاب خاصی پیش نمی‌ریم ☹

بخش دوم


Designing
Data-Intensive
Applications
THE BIG IDEAS BEHIND RELIABLE, SCALABLE,
AND MAINTAINABLE SYSTEMS

پیش‌نیازها

پیش نیازها

- تسلط به زبان جاوا
- بلد بودن Git
- آشنایی با شبکه، TCP و UDP و کار با آن
- آشنایی با سیستم عامل

قوانین درس

تمرین‌های برنامه‌نویسی

- ۵ سری تمرین عملی
- گروه‌های ۳ نفره (با استثناءِ دو نفره ☹)
- تمرین‌ها تحویل حضوری با ارائه دارند.
- جی‌پی‌تی و هم‌فکری و ... ؟؟ متاسفانه اوکیه! ولی اکیدا توصیه می‌کنم که حتما قبل از GPT خودتون تلاش کنید.
- حس بد بگیریم که زیادی کپی کردید می‌تونیم تحویل نگیریم. ☺

گروهی!

- درس یک کار است، نه رابطه‌ی دوستی! شما مسئول انتخاب هم گروهیتان هستید.
- توصیه: پیش از هم گروهی شدن از هم قول بگیرید و انتظارات و زمان مشترک را مشخص کنید.
- همه باید به همه‌ی بخش‌ها مسلط باشند و باهم تمرین زده شده باشد. به بخش‌های دیگر هم توسط بقیه‌ی اعضا دیده شده باشد و همه به همه‌جا مسلط باشند.
- نمره‌ی یکسان برای همه‌ی اعضای هر گروه
- مشخص بودن فعالیت در ریپازیتوری گیت‌هاب

آزمون میان‌ترم!

- آزمون میان‌ترم تئوری روی کاغذ
- ان‌شاءالله در یک پنج‌شنبه در اواخر فروردین

آزمون پایان‌ترم!

- آزمون پایان‌ترم تئوری روی کاغذ
- با تمرکز به مطالب پس از میان‌ترم

اصل مطلب – بارم‌بندی

- **۵ سری تمرین** – هر تمرین ۳ نمره
  و در مجموع - ۱۵ نمره
- **آزمون میان‌ترم** - ۳ نمره
- **آزمون پایان‌ترم** - ۳ نمره

تیم درس

- علیرضا توفیقی محمدی
- سید امیرمحمد سادات شکوهی

بقیه‌ی تیم درس، بزودی...

اطلاعات درس

https://alireza.dev/teaching/ct1403/

تقویم درس بزودی در سایت درس

کوئرای درس

رمز کلاس: ct1403

گروه تلگرامی

تیم درس رو می‌تونید در تلگرام هم بیابید.

QUESTIONS?

INTRODUCTION TO GO

History of Go

- Created by Google in 2007
- Designed by Robert Griesemer, Rob Pike, and Ken Thompson
- Officially launched in 2009 (open source)
- Goal: Efficiency and simplicity

History of Go

- **2007** – Development of Go begins at Google.
  - *Inspired by issues with C++, Java, and other existing languages.*
  - *Aimed to provide simplicity, efficiency, and strong concurrency support.*
- **2009** – Go is publicly announced as an open-source project.
  - *The first version was released on **November 10, 2009**.*
  - *Featured fast compilation, garbage collection, and built-in concurrency support.*
- **2012** – Go 1.0 is released.
  - *Marked as the first stable version with long-term backward compatibility.*
  - *Widely adopted by startups and cloud services.*
- **2015** – Go 1.5 is released.
  - *Removed C dependencies, making Go entirely self-hosting.*
  - *Improved garbage collection, boosting performance.*

History of Go

- **2016** – Kubernetes, built with Go, gains massive popularity.
  - *Go becomes the go-to language for cloud-native development.*
- **2017** – Go 1.8 introduces a much faster garbage collector.
  - *Companies like Docker, HashiCorp, and Uber start using Go extensively.*
- **2019** – Go modules become the standard for dependency management.
  - *Addressed issues with GOPATH and improved package management.*
- **2021** – Go 1.17 introduces performance improvements and optimizations.
  - *Go becomes one of the most loved programming languages in developer surveys.*
- **2022** – Go 1.18 introduces **Generics**, a long-awaited feature.
  - *Greatly enhances code reusability and flexibility.*
- **2023-Present** – Go continues evolving with new features and optimizations.
  - *Popular in cloud computing, microservices, and DevOps tools.*

Why Go Was Created

- Compiles quickly (C++ was too slow to compile at Google’s scale).
- Has simple syntax (unlike complex C++ templates).
- Handles concurrency efficiently (better than Java's threading model).
- Has garbage collection (to avoid manual memory management issues).
- Is safe and easy to maintain (compared to C).

Key characteristics

- „Less is more“
- Open-source
- General purpose
- Procedural, C-family
- Compiled
- Statically typed
- Type inference
- Strict formatting rules
- Allows cross-compilation
- Garbage collected
- Built-in concurrency

Current Use Cases of Go

- Cloud services (Kubernetes, Docker, Terraform)
- Web development (Gin, Echo, Fiber frameworks)
- Distributed systems (gRPC, NATS, etcd)
- Networking tools (Caddy, Traefik)
- Data processing (InfluxDB, Prometheus)

IDEs and editors

- JetBrains GoLand
- VS Code
- More: https://go.dev/wiki/IDEsAndTextEditorPlugins

Installing Go

- The official download guide lets you download:
  - *.msi file for Windows*
  - *.pkg file for MacOS*
  - *.tar.gz archive for Linux*
- *The .msi and .pkg can be executed directly*
- The tar archive has to be unarchived, moved and the path to it exported manually
- $ rm -rf /usr/local/go && tar -C /usr/local -xzf go<version>linux-amd64.tar.gz
- $ export PATH=$PATH:/usr/local/go/bin
- https://go.dev/doc/install

go.dev/doc/install

Documentation > Download and install

Download and install

Download and install Go quickly with the steps described here.

For other content on installing, you might be interested in:

- Managing Go installations -- How to install multiple versions and uninstall.
- Installing Go from source -- How to check out the sources, build them on your own machine, and run them.

Download (1.23.6)

Go installation

Select the tab for your computer's operating system below, then follow its installation instructions.

Linux Mac Windows

1. Open the MSI file you downloaded and follow the prompts to install Go.

   By default, the installer will install Go to `Program Files` or `Program Files (x86)`. You can change the location as needed. After installing, you will need to close and reopen any open command prompts so that changes to the environment made by the installer are reflected at the command prompt.

2. Verify that you've installed Go.

   1. In **Windows**, click the **Start** menu.
   2. In the menu's search box, type `cmd`, then press the **Enter** key.
   3. In the Command Prompt window that appears, type the following command:

      ```
      $ go version
      ```

   4. Confirm that the command prints the installed version of Go.

Go Tour

- https://go.dev/tour