Pennymac

Pennymac (NMLS #35953) is a top national mortgage lender with over 4 million lifetime customers.

and this is because they've been fully in the cloud and serverless-first for many years now.

Our mission is to build a foundation of homeownership by enabling our customers to achieve and sustain their aspirations of home.

My Projects

Health care system

REST API

GraphQL

Message based payment system

Cross chain NFT bridging system

Blockchain Orchestrator Layer

Angular Experience

Angular 6	03-05-2018
Angular 9	06-02-2020

The Angular is a development platform for building a Single Page Application for mobile and desktop, using Typescript & HTML. I've got *familiar with those basic concepts and Angular building blocks like components, Templates, Directives, Data Binding, Forms, Pipes, HTTP Services, Dependency Injection, etc.*

Tell me about yourself.

*Sure, so, my name is ** and I am 30 years old.*

For the past 10 years, I've been working as a software engineer at Company X and Y.

I have got background in full stack development, mainly focusing on the backend part including blockchain and smart contract.

Mostly I've been involved in designing and developing REST APIs for high volume clients with modern development stacks like Go and NodeJS, particularly Golang, getting familiar with Golang development tools and frameworks.

And sometimes I worked with Kafka and RabbitMQ as the message bus and data pipeline layer.

Also having experience with various AWS and GCP services such as …

Feel comfortable working with container and container orchestration technologies such as Docker, Kubernetes, GKE and Amazon EKS.

Ok, moving forward, I hope to expand my experience across different industries, mostly backend oriented,

looks like the opportunity you provide allows us to work on this, which is why I'm interested in your company.

Kafka Use Case

Training course

Activity tracking in employee training course

We had to make sure the employee properly covers all those training materials as requested. So, we needed a system to track all of those users' activities throughout the training course such as user clicks, registrations, likes or favorites, time spent on certain pages, orders, environmental changes and so on. It was often very high volume, as each user page view generates many activity messages or events.

We had a frontend React applications in which these events are published (produced) to dedicated Kafka topics. Also, we have designed and implemented a separate backend application, written in Go, to subscribe to topics, receive the data, and process it to monitor if there are any illegal user activities.

In this system, I have been involved in setting up the kafka server and cluster as well as the whole backend system for kafka consumer part and kafka producer integration in the frontend side.

Blockchain Orchestrator Layer

We've created our own blockchain platform. In this system, we had to handle the user transaction like token deploy, mint, transfer and burning. The system consisted of several different layers like business service components, A&A layers and BCL layers.

I was responsible to build the blockchain orchestrator layer.

The assumption was that the BCL layer uses an event-driven architecture and its interface is through Kafka/Red Panda topics. However, all business service components use a message passing approach. Thus there is a need for a layer to translate to the BCL.

BCL works as an async, event-driven layer, using Redpanda, a Kafka-like broker. That means that Nazgul requests to BCL will be sent to the `transactions.requested` *topic, and responses will be sent to the* `transactions.events` *topic.*

Company

Why did you decide to apply for this position?

It's simple, I read your company's mission and vision. After finding out more information about your company from social media, I can clearly see how my goals align with the purposes of the company. There's nothing I love more than when I am working with other engaged individuals towards a common goal, and that's what excited me about working here.

Why do you prefer contract role?

I suppose contracting is a smart way to broaden your experience and diversify your skills gaining exposure to a wide range of companies and industries

Proactive & Reactive

What would you do in your first month of working here?

Along with completing my training, I hope to make positive connections with all of my clients within the first month of working here. I want to use this first month to prove to clients that I can do an excellent job at serving their needs.

Tell me about a time you took initiative.

When working for my previous employer, I noticed that we didn't have a good system for keeping track of client work. We actually did everything in a giant spreadsheet, which I felt wasn't the best system. That's why I took it upon myself to find a project management system that suited our team well. It took me about a month of searching and doing trial runs to find the perfect one. Once I implemented this system, my team members and clients were thrilled that we had a more user-friendly system and better-looking interface.

Tell me about your solid example of proactive engagement

Proactive leaders (and their teams) are confident in their ability to foresee and plan around potential challenges.

When difficulties arise, the response time is often shortened by having a strong plan in place.

When working for my previous employer, I noticed that we didn't have a good system for keeping track of client work. We actually did everything in a giant spreadsheet, which I felt wasn't the best system. That's why I took it upon myself to find a project management system that suited our team well. It took me about a month of searching and doing trial runs to find the perfect one. Once I implemented this system, my team members and clients were thrilled that we had a more user-friendly system and better-looking interface.

Tell me about your solid example of reactive engagement

Being reactive means that you react to situations through your emotions. Here, you can often come across as blaming, resentful, insecure, or angry.

Reactive people are often more comfortable with uncertainty and trying different approaches.

Ex; Different approaches for message passing protocol

My Experience

What is the most challenging engineering project you dealt with? How did you ensure it was a success?

I was once tasked to implement the cross chain bridge service in Go while working for MerkleRoot company.

I had to use Go's concurrent support in this project, you know, concurrent handling is one of the most powerful function Go supports in its language level. But I found it's really tricky and complicated to understand and make use of it.

I had to get through understanding its core concepts, theories and technologies under the hood. I tried to cover the useful samples and also asked my colleagues and project members for help. Finally I was able to understand it and build more efficient and flexible source codes.

What processes do you use to make sure you haven't made any mistakes?

When I'm working on a project or assignment, I take several steps to ensure there aren't any errors in the end result. First, I always recheck my work as I go, using both a visual review and software tools designed to identify issues.

Second, I'll get insight from a trusted colleague or other project members. At times, a second set of eyes can spot issues you may have accidentally overlooked or that the software misses. While problems at this point are rare, I find that it's a practical step for ensuring accuracy.

Finally, before any work is turned in, I give it another review. Ideally, I set the project down for a moment, even if it's brief. That way, when I return, I can look at it through fresher eyes, increasing the odds that, if there is an issue, I'll spot it.

What steps do you take to keep your engineering skills current?

Maintaining and augmenting my skill set is very important to me. As a result, I've embraced several approaches to ensure I stay on top of new developments.

First, I am a member of a professional engineering association. Not only do they share industry news, but they also host conferences where new developments are discussed.

Second, I follow industry leaders on social media, belong to several email newsletters, and have a curated RSS feed. This keeps me apprised of recent headlines and intriguing trends.

Finally, one of my most important steps is continuing education. Once an emerging trend begins to take hold, I seek out formal educational opportunities focused on the subject. That way, I can get my footing early, ensuring I'm ready the moment my employer decides to move forward with it.

What is your greatest achievement?

My greatest professional achievement was turning around the success of my last employer's Marketing department.

I was working as a software engineer, not as a marketing engineer by the way.

At that time, the entire team was struggling and we were failing to hit our quarterly goals.

The employer asked us to help him create a new marketing plan, which I designed without any guidance.

Within 6 months, we were achieving 20-25% above our goals, and my marketing plan brought in an additional $500K in revenue for the company through the second half of last year.

This additional revenue brought the company from an operating loss each quarter to profitability.

What is your greatest strength as a software developer?

*I'm able to give accurate project estimates. My supervisor at ** worked with me on this so that I learned how to do it well. Then at my last company, I became the person who checked everyone else's estimates before they sent them off to our supervisor.*

What is your greatest weakness as a software developer?

I tend to spend too much time on some tasks and not leave enough for others, so I created a calendar where I can block out when I'll work on which project. I set timers to go off in the middle and end of each block so that I know when I need to start wrapping it up.

Agile experience

As a member of an agile engineering team, we opened a weekly planning meeting to design the application and assign tickets to team members.

We also opened up a daily sync to see if there are any blocking issues and how the development process is going for each one.

What's important when checking a team member's code?

1. *Don't rush your review*
    a. *Limit your review to 200-400 lines of code at once*
    b. *Don't spend more than one hour at a time reviewing code*
    c. *Plan for 3 seconds of review time per line of code*
2. *Provide constructive feedback*
3. *Use a code review checklist*
    a. *Readability*
    b. *Performance*
    c. *Reusability*
    d. *Maintainability*
4. *Use code review tools*
    a. *GitHub, GitLab, BitBucket*
5. *Focusing on coding standards, not personal preference*

Questions to Ask at the End of an Engineering Interview

- *What are your expectations in the short term from me?*
- *What is the business reason for this project?*

- *What would you task me with first?*
- *What long term goals do you have for the company?*
- *What would be important to be successful in this role?*
- *What do your most successful engineers in this role have in common?*
- *If you had one piece of advice to give the new hire in this role that would propel them toward success, what would it be?*

Go Technical

Why Go?

Go is rapidly gaining popularity as a web development language.

Many companies, including infrastructure companies like Dropbox and SendGrid, technology-oriented

companies such as Square and Hailo, as well as more traditional companies such as

BBC and The New York Times, have already started using Go.

Go provides a viable alternative to existing languages and platforms for developing

large-scale web applications, providing Scalable, Modular, Maintainable and High-performance.

One of Go's design goals is to approach the performance of C, and although it hasn't reached this goal, the current results are quite competitive.

Go concurrency

Golang is a very efficient language. Much of this efficiency can be due to its unique concurrency abstractions. For example, Java threads are mapped to OS threads, whereas Go utilizes its own goroutines scheduler to further insulate its lightweight goroutines from OS threads. In summary, Golang uses OS threads sparingly; if a goroutine *becomes stuck, Go's scheduler will replace it with another goroutine to keep the thread occupied as much as possible. Because each CPU core can only handle a certain number of threads (and generating new threads is expensive), keeping these threads busy is essential.*

So, how can we create several http calls in Golang at the same time? You may have used async/await to make several API requests if you've used C# or current JavaScript. It's not quite as simple with Golang, but it's all in the name of efficiency. At least one goroutine is constantly running in Go, which is responsible for running the main program (). With the term go before the function call, we can spawn new routines. Goroutines may remind you of the concept of context if you've worked with Java/C# asynchronous calls. The developer can create thousands of these lightweight goroutines with Go Scheduler, and it will manage the CPU time spent on each one for us.The main goroutine continues on its path immediately after spawning a new goroutine, until it reaches a blocking operator (akin to an await in C# or Js).

Go Unit Test

https://github.com/stretchr/testify

Testify is an incredibly popular testing framework. It makes available a range of assertion functions for comparing and verifying values. It also offers mock functionality to easily mock resources. Its suite package aids in creating test suites for grouping related tests, along with offering setup and teardown functionality for test preparation and cleanup.

The assertion and mock functionalities make testing easier and faster. With test suites, you can collect related tests and create a shared test setup and teardown. Testify's test output is standard and not as detailed. However, it does offer the option to annotate assertions with messages, making the tests more expressive.

Testify is very user-friendly. It works well with the testing package and go test command. It does not offer its own custom test coverage reporting, unlike other packages. However, coverage can be obtained through the testing package. Its pkg documentation page is thorough and offers numerous examples.

There is a large community of users and contributors for Testify, but updates and new features are few and far between. Testify does offer multiple Slack channels where its users can interact and seek help, so finding support is fairly painless.

Go profiling / benchmarking

Profiling is measuring the performance of your program in various dimensions. Go comes with great support for profiling and can profile the following dimensions out of the box:

- *a sampling of CPU time per function AND instruction*
- *a sampling of all heap allocations*
- *stack traces of all current goroutines*

- *stack traces that led to the creation of new OS threads*
- *stack traces that led to blocking on synchronization primitives*
- *stack traces of holders of contended mutexes*

You can even create custom profiles if you want. Go profiling involves creating a profile file and then analyzing it using the pprof go tool.

General Technical

What are design patterns?

Design patterns are the reusable solutions that solve common problems of software development. These problems include repetitive code, redundant functions and logic etc. These help to save considerable effort and time required for the developers while developing software. Design patterns are commonly used in object-oriented software products by incorporating best practices and promoting reusability for developing robust code.

What are the types of design patterns?

Creational Patterns*: These patterns provide freedom of choice between creating objects by hiding the logic. The objects constructed are decoupled from the implemented system. Some of the examples of creational patterns are - Factory design pattern, Builder design, Prototype design, Singleton design, Abstract Factory design.*

Structural Patterns*: These patterns help in defining how the structures of classes and objects should be like for defining the composition* *between classes, interfaces and objects. Some of the examples of structural patterns are - Adaptor design, Facade design, Decorator design, proxy design etc.*

***Behavioural Patterns**: These patterns help to define how the objects should communicate and interact with one another. Some of the examples of behavioural patterns are - Command pattern, Iterator pattern, Observer pattern, Strategy pattern, etc.*

What is Inversion of Control?

Inversion of control is a pattern used to decouple the dependencies between layers and components in the system. The Dependency-Injection (DI) pattern is an example of an IoC pattern that helps in removing dependencies in the code.

Deploy microservice on Kubernetes

Create microservice to be deployed

Placed application in your docker container

Create a new Kubernetes project

Create new Cluster

Allow access from your local machine

Create, activate service account

Connect to cluster

Gcloud initialization

Generate access token

Deploy and start Kubernetes dashboard

Deploy microservice