# Welcome

Welcome to the WealthForge development team!

In front of you is a rare opportunity. In 2015 there were about 7 million open job positions for some form of computer programming[[1]](#footnote-0). Every year in the US there are about 30,000 computer science degrees awarded to students[[2]](#footnote-1). That number doesn’t include the number of hobbyists and self taught programmers. Despite the enormous demand for and supply of people capable of programming, most software created is narrow, specific, and internal.

It’s a very rare thing to get to work on software that is customer facing and truly changes how an industry functions. - You get to do that at WealthForge

The buying and selling of stocks has been automated for some time now. Etrade, Scottrade, Vanguard, etc. have made it easy to set up accounts, and invest in publicly traded securities. Kickstarter has made it easy for individuals and small institutions to raise money through private donations to an effort. However, there is a vast middle ground between publicly traded securities and gifts where debt and equity are traded. This middle ground is largely opaque, complex and forbidding to most investors.

We’re gonna change that.

The buying and selling of equity in privately held companies is a highly regulated market because it is ripe for con artists and scams. It is the market where deals are not public, and reporting is much more relaxed. This does not mean that the market is shady or bad. The vast majority of deals are legit. But privately held companies don’t receive the same scrutiny or have the same reporting requirements as publicly traded companies (for many, that’s the whole point of being a privately held company!). So when a private company wants to raise money by selling equity, or debt to fund a project or to grow, they have a complex battery of reporting and diligence they must meet in order to issue securities to investors. This hurdle hurts the market because it essentially increases the cost of capital for private companies. Additionally, investors who may be very excited about investing in a certain project may never know that an opportunity exists because it’s not public.

WealthForge is different because we seek to streamline this process by guiding issuers through the process of offering private securities, making this information available to investors, and automating the due-diligence process for all parties.

# Business and Nomenclature

In order to build the right software for WealthForge and its customers, you first need to understand the business. This section will guide you through the business specific vocabulary and lay out how business processes function.

## Nomenclature

**Offering** - This is the center of our universe. Offering is the investment opportunity in which one party sells debt or equity for cash.

**Sponsor** - This is a company that coordinates all the work around raising capital for large projects. This may include making introductions to the right people, inspecting properties and coordinating work around fundraising. These entities are usually serial fundraisers. They raise money for multiple other companies all the time. These companies are not usually FINRA regulated.

**Issuer** - The entity who is issuing debt or equity to raise capital. This is usually a private company trying to raise money so they can grow and/or continue to operate.

**SPV -** SPV is Special Purpose Vehicle. This is a separate legal entity set up by a sponsor to issue securities for a specific project. The business purpose is to insulate the Sponsor (the company that wants to raise capital) from any legal kerfuffles that could arise from the attempt to raise money. An SPV is always an issuer, but an issuer is not always an SPV. On the tech side of the house, we generally just use Issuer since this term covers everything we care about from a technical/process point of view. SPV is just called out here so you know what it means when you hear it.

**Subscription** - This represents the commitment to invest in an offering. It’s essentially when an investor says, “here’s my info. I’d like to buy in to this offering.” It’s not a full investment yet, because a good deal of diligence is conducted before the subscription is approved. This is a laborious and manual process.

**Subscriber** - A person who has issued a subscription to invest in something. This is the person or company that has issued a subscription to buy into an offering.

**Investment** - This is a subscription after it’s graduated. This means that some private individual or company now has ownership stake in the company that was trying to raise money.

**Investor** - This is someone who wants to or does invest in an offering. This person/business buys debt or equity from a sponsor.

**Broker/Dealer** - The company that brokers the deals (buying and selling securities). This is a FINRA regulated role.

# Communication

We have the great fortune of being colocated which means most of our communication is done in person. We expect that every effort will be made to attend sprint planning, retrospective, and daily standup meetings.

Outside of actual face-to-face talking, we use Slack as our chat client. There are three main channels that developers should be in. The main channel that you should be interested in is **DevMain** - this is the channel for all technical discussion that relates to our product. After that, be in **General** channel - this is for all random chat and memes. Our meme game is fairly weak, and we could use some help in this area. Finally, there is the **Personal Status** channel - this is just for letting the team know when/if you’ll be late of out of the office.

For tracking and communicating work progress, we use Rally. This lets us see progress toward our overall software goal, but also lets us break down work to very manageable tasks that the engineering team can effectively discuss and complete.

# Best Practices

## Agile/Scrum

Our team uses a textbook Scrum method for prioritizing, estimating, and completing work. This method provides a good amount of protection so that developers get long blocks of uninterupted time to do their work, but is still flexible enough to respond to the changing priorities of our leadership and sales teams.

Our backlog is made up of prioritized User Stories. A User Story is the smallest amount of work that has value to the customer/user. Our goal is to take all the work necessary for the creation and maintenance of our product and break it down to chunks of work that are manageable for the developers.

Our team has decided that our User Stories must have the following:

### User Story

The User Story should have a statement that lets the engineer know who wants the functionality, exactly what they want, and why they want it.

For example: *As a WF portal user, I want to be able download offering information in PDF format so that I can share it with my customers in both electronic and paper form.*

This lets us know who wants the functionality, what the functionality should be, and why they want it. The work should be as small as possible, while still providing value to the user. Obviously, the functionality described above could be broken down further, but this is the smallest amount of work that still provides value to the WF user.

### Acceptance Criteria

Every User Story must have an acceptance criteria (sometimes called a ‘definition of done’). This is a statement that serves as a binary litmus test for whether a User Story is completed or not. The Usert Story statement gives us purpose, but the acceptance criteria should provide unequivocal understanding of whether something is done or not.

Often times, the acceptance criteria may seem like a rehashing of the User Story.

Ex.: the user is able to download offering details in a PDF.

However, most user stories are no so clear cut. For example, if we get feedback that “the login process is too ‘clicky’ and I can’t find what I need.” We might generate User Stories with the following acceptance criteria:

Bad: Make the login process less clicky

Good: Users are able to log in with three clicks or fewer

Bad: Users can find their info quickly.

Good: From the landing page, users can access profile info, offering information, client info, or broker dealer info with a single click.

### Task

A user story may take more than one engineer to complete. It is governed by value to the user/customer. That is to say, it may be bigger than we like, but it is designed to capture business value. A large User Story may require the work of three different engineers because it touches on three or more skillsets to deliver the value.

A task is different. A task is owned by only one engineer. A task is the smallest logical unit of work that advances our work toward completing a User Story. A large User Story may have dozens of different tasks.

For example, for the following single User Story: *As a WF portal sponsor, I want to be able to create an offering so that I can list new investment opportunities for others to see.*

That statement captures a functionality that has value to our customers. However, the tasks involved with completing this might looks like this:

* Gather offering page use cases
* Create offering page wireframe
* Create offering page mockup
* Define offering page fields
* Map offering page fields to database fields
* Create offering page
* Verify offering page API functionality
* Create integration tests
* Create functionality tests
* Document integrations

The task level is where developers spend most of their time. It makes it easier to communicate problems and share work if we all have an understanding of how the engineers are completing their User Story. It also provides a roadmap for each User Story. This way if an engineer is out sick for a week, anyone following behind them can get an understanding of what the plan was for finishing the story and what work has already been accomplished.

### Wireframes/Mockups

We make a customer facing product which means that we have to consider the appearance and flow of our product from the customer’s point of view. For any User Story that is customer facing (and many that are not), we expect there to be a wireframe and mockup of how the functionality should look and perform.

A wireframe an image or set of images which displays the functional elements of a website or page, typically used for planning a site's structure and functionality.

A mockup is a model or replica of a machine or structure, used for instructional or experimental purposes. For us, this is primarily used to show what a webpage should look like so that it’s easy to use and understand for our customer.

### Sprints/Timeboxes

At a high level, a project is governed by one of two things - time or quality. For example when I tell my son to clean his room, I don’t really care how long it takes. That project is governed by quality. He’s not done until the room is clean (as defined by me). Most projects are governed by time. This is where the idea of the deadline comes from. Of course we want quality, but the product is due by 3:00. This is where we hear things like “Do the best you can,” implicit in this statement is the idea that you should do the best you can in the time you have.

At WealthForge, we expect our development work to be governed by quality.

Let’s say that again -

At WealthForge, we expect our development work to be governed by quality.

To help us create quality software while still respecting time constraints, we break our work down into time boxes (sprints). These time boxes should protect the developer from random requests or frequent changes in priority/work. At the start of the sprint, we figure out how much work we can take on over a two week period. We accept work into our sprint, and after that we are mostly protected from additional work. This gives us time (two weeks) to do quality work without interruption. At the end of the two weeks, we review what we’ve done, and prepare for the next two weeks.

## Naming Conventions

Part of being on a team means doing work that is easily understood by your teammates. It is incorrect to think that we are writing code for machines alone to understand. All longstanding software will be read and re-read by multiple people, multiple times. To make this easier for all the people who follow behind us, we have come up with some basic best practices for naming

## Professionalism

Everything written will be professional. This means emails, chat, code commits - Everything. We have fun at the office. We goof off and joke around. We will be candid with criticism and concerns. But we have to assume that there are forces at work out there that are looking for anything disparaging to say about WealthForge or its people. The only way to counter these malevolent forces is to stay professional in all written communications. Right Andrew?[[3]](#footnote-2)

1. <https://www.fastcompany.com/3060883/why-coding-is-the-job-skill-of-the-future-for-everyone> [↑](#footnote-ref-0)
2. <https://datausa.io/profile/cip/110701/> [↑](#footnote-ref-1)
3. Ask Batz about this! It’s a fun story. [↑](#footnote-ref-2)