Mark Felchlin

Introduction

I love to play volleyball and love to go exploring outdoors. I am was raised in Spokane since I was one, but I was actually born in Augusta, Georgia. I lived in Santa Barbara for 7 weeks coaching beach volleyball. I did running start here at EWU for two years which ultimately lead me to stay and the fact that it is the cheapest four year degree in the state and close to home. I have always loved technology and been decent with computers, my father recommended it so I tried a class. After the first class I fell in love with it and ran. I will be using Windows as my operating system of choice for the course.

Research Topics

The front-end developer is responsible for creating complex UI and UX systems. Front-end develops have extensive knowledge of HTML and CSS. Often these developers also have knowledge about tech stacks that are based off the JavaScript framework like Vue, Angular, and React. You commonly run into the front-end developers that majored in some area of design and a lot of them do not actually get a major in computer science. On the other hand, a development team will have a back-end developer. Back-end developers are skilled and knowledgeable with the manipulation and storage of data using database systems like SQL, MongoDB, Azure Cosmos, and SQLite. These developers are knowledgeable about the ideal architecture of a software system and the best practices to pass data around efficiently and securely. Often, they also work hard at securing an application and making sure their data is safe.

The MVC model breaks down an application in to three parts that work together to transfer needed data. The idea behind this pattern is to enable everything to only know about what it needs. What is displayed does not need to know how you connect to the database to check the user credentials before allowing them to go to the next page. The model is where all the logic happens, if the user attempts to log in to the system then the model will take the data from the view and validate that what the user gave and then return something. Now what the View gives us is not something that the model is able to understand all the time, so this is where the controller comes in. The controller takes whatever the View gave it and then turns that into some command or data for the Model to do something with it. Once that is done the controller

takes the data and manipulates the view to display whatever based on the data from the Model. When it comes to the View its sole purpose is to display stuff and give user input to the controller so that business logic can happen. This model is wonderful because we are able to break out responsibility and makes the software easier to scale and grow.

When it comes to IDEs it really depends on what exactly you are developing. If you are developing for Mac then you most likely prefer to use XCode. XCode specializes in Objective-c and Swift development. Now if you are developing a .NET program you will want to use Visual Studio as it is seen as the best for WPF editing. For developing Java, the most common IDEs are Eclipse, VS Code, and IntelliJ. I personally love to use IntelliJ for development as it feels the most comprehensive and simple while also being fast. Sublime Text and WebStorm are also a couple of notable IDEs commonly used for web development.

The active record pattern is an architectural pattern that stores in-memory object data in relational databases. To start with the pros, it is great for domain logic that is not very complex. It is great when a user is only wanting to do a very small task and there is very little business logic involved. On the other side of the coin it violates the single responsibility principle and attempts to do everything in one area. Also, you are directly interacting with the database which can cause problems when people forget, and it messes up the data.

PHP Links

https://www.youtube.com/watch?v=OK_JCtrrv-c

https://www.youtube.com/watch?v=pWG7ajC_OVo&list=PL4cUxeGkcC9gksOX3Kd9KPo-O68ncT05o

https://www.youtube.com/watch?v=oJbfyzaA2QA&list=PLillGF-Rfqbap2IB6ZS4BBBcYPagAjpjn

Laravel

https://www.youtube.com/watch?v=ImtZ5yENzgE

https://www.youtube.com/watch?v=ubfxi21M1vQ

https://www.youtube.com/watch?v=BXiHvgrJfkg

SQL

https://www.youtube.com/watch?v=HXV3zeQKqGY

https://www.youtube.com/watch?v=7S_tz1z_5bA

https://www.youtube.com/watch?v=Et2khGnrIqc&t=702s

Front-end frameworks exist to keep the state of the UI synced with the state. Frameworks like Vue, React, and Angular all help organize a UI into a bunch of components and enable a programmer to perform business logic on the client side. React was built by Facebook and a community of developers. React is only a part of what you would need to be able to build an full application as it focuses on working with a virtual DOM, for state management and routing you are going to have to look elsewhere I am afraid. Next, we have Angular which was built by Google with the help from so other developers as well. The interesting thing about Angular is that it is a lot like a framework, and you can manage state and routing without having to use another technology like React does. The last of the three core JavaScript frameworks out there is Vue, which was created by a guy named Evan You and is maintained by the open source community to this day. Vue is a lot like React and it focuses on declarative rendering and component composition. Also, like its counterpart React it looks to third party libraries for state management and routing.