How to Succeed in Bethel Tech

Things you were probably told, but may have missed.

1. Bootcamp: Non-traditional Schooling

Welcome to Bethel Tech! As a coding bootcamp, in many ways we are unlike traditional education environments you have experienced in the past. Our main goal is to teach you how to **THINK like a coder**, not only learn coding languages. This is because the mentality of **creative problem solving and troubleshooting** does not change even as languages and technology do.

We have set up our coding bootcamp to help you become a developer. Bootcamps by definition are not meant to be easy. They are structured to break down what you are used to in order to **teach you a new way of thinking**.

That being said, make a promise to not be so hard on yourself when you don't get something right away! Approach this the way you would if you were to learn Russian or ancient Hebrew: **it takes time** to become familiar with a new language, a new way of thinking, a new alphabet. This is a full immersion experience, like being dropped into St. Petersburg to learn Russian by doing homework in that language. The only way to learn it is by **daily practice** and allowing yourself the freedom to follow your curiosity, make mistakes, and explore what you do not already know.

We believe that you are here because **God called you here** and you have an innate, God-given ability to learn new things and overcome challenges with Him. Take a moment to **connect to the reason WHY** you came here in the first place. Write it down, post it everywhere, and come back to it when you face frustration or roadblocks. **You CAN do this**!

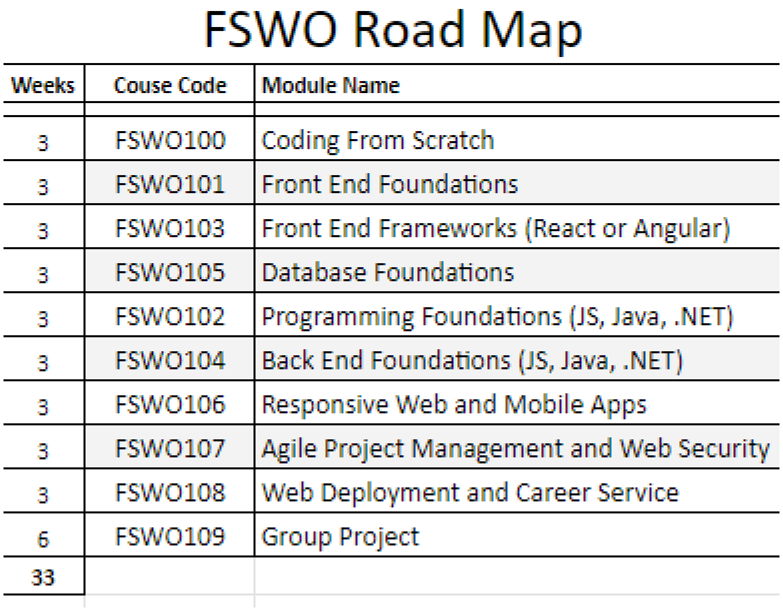
This document is a result of the distillation of the experiences and wisdom of many BST graduates and staff members. Take time in the first few weeks of school to read and do the things listed. Here are some helpful ways to make sure you get to your goal!

2. Prioritizing

- **Know Your Needs**: Are you a very social person? Do you feel refreshed after going on a walk? What makes you come alive? It will help you immensely to understand what you need while you are in this 9-month intensive period of learning so you can recognise when you need to take a break to do what refreshes you and when you need to keep pressing in.
- **Priorities**: You can do anything for a season, but it takes reprioritization to make that happen. **Take a look at your previous commitments** and **reassess if you are able to continue** with the volunteer ministry, opportunities, and social activities in your life. You might **need to say no to some very good things** for a short period of time so your Yes can be big to learning this new skill.
- **Come to an Agreement with Your Spouse or Support System**: If you are married or closely work with your family, in order to be as successful as possible, **be clear about what it will take** to make this happen. Recognize what sacrifices you both are willing to make for 9 months in order to complete this program.
- **First 3 Modules**: Realistically, **your first three mods will be the toughest** to get through. Choose now to give yourself grace when you

don't understand, to keep yourself going when you want to quit, and to always get back on the horse if you get bucked off. Allow yourself extra time to soak up what you're learning and navigate frustrations and roadblocks. **Do not rush**! Rushing through things shuts down your agility and adaptability, and it prevents you from noticing the little details that get missed when you try to hurry. Hurrying ends up hurting!

- Here is the **program road map** of all modules to help you keep on track:



FSWO Road Map

Weeks	Couse Code	Module Name
3	FSWO100	Coding From Scratch
3	FSWO101	Front End Foundations
3	FSWO103	Front End Frameworks (React or Angular)
3	FSWO105	Database Foundations
3	FSWO102	Programming Foundations (JS, Java, .NET)
3	FSWO104	Back End Foundations (JS, Java, .NET)
3	FSWO106	Responsive Web and Mobile Apps
3	FSWO107	Agile Project Management and Web Security
3	FSWO108	Web Deployment and Career Service
6	FSWO109	Group Project
33		

3. Time Management

There are plenty of tools to help you get a handle on the way you manage your time. Sometimes it can feel overwhelming to navigate through so many choices, so we will list a few here, and we recommend that you pick one of them and develop it!

- **Google calendar**: This is a great option as it will sync across your devices. Set up repeating time blocks on your calendar for your work/commute hours, time you spend with your family, your main commitments, and the times you set aside for coding. You can set up reminders in each time block to alert you to when it is time to sit down and code.
- **Trello**: If you think task- or project-based, Trello is for you! You can create columns for the projects you are working on, and individual cards for the tasks you need to get done. You can even set up a system of "To Do, Doing, Done" that lets you see your progress in real time and celebrate your done list.
- **Reminders**: A lot of our students have found success with setting up reminders on their phone to let you know what modules you should be

working on and when they are due. You can also set a daily reminder for the time that you sit down to code.

- **Planner**: If you process information best by writing it down, it could be helpful for your brain to draw or write out what your week will look like and what you need to get done. It could be a good balance to all the computer work you're doing!

4. Resourcing Yourself

- **Researching**: **Google** and **YouTube** will become your **best friends.** Even as you become a developer, you will constantly be engaging the developer community online. Learning and gleaning from developers around the world will help you become a high-level **creative problem solver** and allow you to stand on the shoulders of giants in the development world as you learn to build and code. Our program is **set up this way on purpose**, so set yourself a goal to become a **constant researcher** and **excellent question asker**!
- **Get a Grasp On How**: One of our coding mentors, Darrin, gave the example that if he is working on a new project and has 2 hours to code, he will spend **an hour and a half researching** and use 30 minutes to code. This is not a formula, but it does highlight the importance of researching what you're working on, how others have worked on solving that before, and getting a grasp on **how to do what you're about to do**.
- **How to Learn a New Skill**: Check out this quick article about **10 ways to help you learn a new skill** (i.e. coding!). These will set you up for success as you implement them in your approach to something completely new. We believe that ANYONE can learn how to code, and these steps are jumping off points to success.
    - "**Top 10 Strategies for Learning New Skills**": https://zapier.com/blog/learning-new-skills/

5. Reach Out for Help

- **Over Communicate**: We want to encourage you to **OVER COMMUNICATE when you have questions and are stuck**. As you are plunging into this new world of coding, our online community of BST students is here to help you!
    - **Reach out right away** in as many channels as is appropriate. If you don't feel annoying, you're probably asking enough questions. Sometimes the articulation of the problem you're experiencing helps you gain a new level of understanding, and the more detail you can provide, the more you enable others to help you.

    - There is **grace** for those who communicate, but those who don't will miss out on creative solutions, opportunities, and building connections with others.
- **Be Proactive**: Don't just wait around when you're to get help or an answer. Be proactive in seeking out solutions and help. Contrary to what some might think, you will not be able to do this program alone. Make connections, build relationships, and send DMs to new friends to see if you can figure things out together.
- **Slack**: Slack is an incredible tool that facilitates learning, connection, and sharing. You can group message several people at once, you can direct message your coding mentors or friends, and you can post in group channels where lots of people are sharing information and gifs :D
    - Make sure to **format your username as your full first and last name**, it makes finding each other much easier!
    - To find new channels in Slack, you can **click on the word Channels** on your side bar to discover all the available options for you to join. There is **one for each module**, so check back each time you progress to a new class!
    - Check out this website for some great intro articles if you're using Slack for the first time: https://get.slack.help/hc/en-us/articles/218080037-Getting-started-for-new-members
- **Code Together**: If you're looking for another level of help, take the initiative to set up **group coding sessions** with your fellow students. Use your **Zoom** link to create a room where multiple people can connect and code together. Even if your group has students in multiple modules, the cross-pollination actually **increases your ability to learn** new things as you help each other and cement learning in areas you've already gained a level of understanding as you teach others.
    - The more you work in a group as you go, the more set-up for your Final Group Project you will be as you get to the end of the course.

How to Ask Stellar Questions

One of our graduates wrote this beautiful outline of **how to best frame your questions** to get the answers you're looking for. As you go to Slack to troubleshoot the issues you're experiencing, take this outline and use these formulas to ask powerful questions.

- Here are some **practical ways to ask for help** to increase the **speed** and **effectiveness** of the question / answer process.
    1. Be specific
    2. Give points of reference

3. Use keywords
4. Explain the problem
5. Describe expectations vs reality
6. Provide relevant code snippets

Below are some examples of post styles and a bit of why they work or definitely do NOT work:

“I need help”, “can someone help me”, “is anybody on?”

- On its own, these statements do not specify what area of help is needed, and therefore make it hard for a reader to determine if they will be able to help you. It also creates a large lag time while you go back and forth. This **will not get you the help** you are looking for.

“Can someone help with Lesson 3 hands on”

- This is a slight improvement from above, but **does not help readers** know what the question is about. There is a Lesson 3 Hands On for every module, so you will need to be more specific. This is difficult for readers who also may have had different reference numbers or may be mentoring without curriculum at hand.

“I am working on JS Lesson 1 final which asks you to ____” (create an id card, produce a counter, post 10 photos on a site, build a personal website)

- Now we are **getting clearer**. At this point the reader knows what topic you need help with and can gauge if they have a scope of knowledge that might help. BUT they cant tell here what your specific issue is yet.

“In Lesson x, final where we need to ___ the ____ part is not working” (ex. button, input, counter)

- The reader can now tell **what the concept is** you are working on and the section of that concept that is broken. BUT, they can not dig into why because they do not know your expectations vs reality

“In Lesson x, final where we need to ___ . I have a button. Upon clicking it, I expect _____ to happen. Instead ______ happens” (ex. button, input, counter)

- This is a **concise, explained problem**. The reader might be able to offer suggestions or troubleshoot for you without even looking at code. This is particularly true if it is a common problem or if dealt with a similar situation during their time in the course.

“In Lesson x, final where we need to ___ . I have a button. Upon clicking it, I expect _____ to happen. Instead ______ happens” Here is my code: <Student enters entire code dump onto slack as either a photo, attachment or snippet>

- This is a pretty good post for the first few weeks of the course. A reader sitting on a desktop computer can probably troubleshoot this. As the code becomes more complex, pasting the specific sections related to the area you are working on will be increasingly important as pasting too much code makes it hard to navigate

“In Lesson x, final where we need to ___ . I have a button. Upon clicking it, I expect _____ to happen. Instead ______ happens. I think the problem is on line 6-10, but I am not sure. Here is my code:
<Student enters code as a code snippet>
Note: *You can embed code snippets into your slack post by using a backtick on both sides of the text (its the dash located on the same button as the ~. It looks like a single quote but slants the other way). You can also upload a snippet file.*

- This is a helpful post to a reader and allows for study, reflection, research and feedback. This also allows a reader on their cell phone or on the go to read the code and suggest which line might need attention. A reply could look like: “on line 7 it looks like you might be missing a {.“

6. Code Reviews

- **Code Reviews**: Code reviews are one-on-one meetings with your assigned coding mentor who is fluent in your chosen language. These times are for you to get help with your lessons in your modules. **Code Reviews are required**, and the link to your mentor’s Zoom room is included in the Google calendar invite.
  - This meeting allows you to ask questions about the program, get help with your code, and to make sure you are on track to graduate. **Come with questions and work done** to review with your mentor.
  - Reviews are **30 minutes** long. Mentors have many students **scheduled back to back**, so they may need to end the review before they have answered all you questions.
  - **Honor goes a long way**. Treat your mentor with the same kindness and grace you expect to receive.
- What to do if you are unable to complete your Code Review:
  - Schedule extra time to meet as available.
  - Finish conversation via Slack
  - Use Slack to get help from other students
- How to use Zoom:
  - Zoom is a powerful communication tool we use for video calls and sharing screens. Your coding mentor can see what you are working on and request remote access to give example directly on

your computer. You must be **at your computer** during Code Review times in order to share your screen.

        - To download Zoom, visit zoom.us.

- How to share screen in Zoom:
    - Hover your mouse over the Zoom screen
    - A black bar will appear with a green share Icon in the middle
    - Click the share icon
    - Options for shareable screens will appear: Choose the screen with the mentors picture on it.
    - Click the blue share button in the bottom right.
- How to use Slack:
    - **How to find mentors**: To find a mentor, click on Direct Messages (or use the keyboard shortcut Control/Command+K) and type in "//" and a list of staff and volunteer mentors will appear. You can also type in the mentors' name.
    - **How to share files and code snippets**: There is a paperclip icon the left-hand side of the input bar, and you can drag-and-drop files directly onto the conversation. Some may need to be zipped, but Slack will alert you if that is the case. If you need to know **how to make a code snippet**, view this PDF our mentor made! If you need to know **how to take a screenshot**, view this doc with instructions!
    - **How to find who is in your cohort**: You are automatically added to your cohort channel when you begin. To find out who your fellow classmates are, click on the "person" icon under your cohort channel name on the top left. This will give you a list of all the people included in your channel.

7. When Modules Are Due

- **Due Dates**: Due dates are always on **Saturdays**. There are typically **10 lessons** per module. During your mod, you will complete:
    - *first week*: Lessons 1-4
    - *second week*: Lessons 5-7
    - *third week*: Lessons 8-9 and your Final Project
- **Final Projects**: The final project for your module will always be due on the last Saturday at **Midnight Arizona time**. **If your final due date shows Sunday, this is incorrect.**
- **Due Dates Confirmed**: Your new due dates should appear the first Monday of your module in the upper left-hand side of your Exeter dashboard. One of the Coding Mentors confirms all new due dates, so let your mentor know if they are not there by Tuesday.

    - **Note**: In order to see ALL of your due dates in Exeter, hit the Next Arrow to see the next page.
- **Turn In What You Have**: As you work on your final project and approach your deadline, we recommend you **turn in whatever you have**, **even if it doesn't work**. There have been several cases that students would have passed their module even with broken code. **Turning in nothing guarantees you will fail** your module, but if you just turn in something, you can still pass!

8. What To Do When Life Happens

- **Options**: We get it, life happens. And that's okay! **You have options**. But we can't help you if you don't tell us. Be proactive in your communication, and we can get you set up to succeed.
- **Leave of Absence**: An LOA is to prevent failing your module when you have a pre-planned events that will inhibit you from making progress in your module. If you have an event that is a few days long, we encourage you to get ahead and work hard to catch up. **Proactively connect** with your Retention Specialist to discuss your situation ahead of time.
- **LOA Details**: They are typically **3 or 6 weeks** in length, one or two full lengths of a module. They begin at the start of the module during the time you will be gone, and you will return when the next module begins. They prevent you from failing a module, but they also extend out your graduation date 3 or 6 weeks as well. You are allowed to take a **maximum of 6 weeks** while you are at Bethel Tech.
- **Emergencies**: If you are experiencing a personal or family emergency, contact either your Spiritual Mentor, Retention Specialist, or Enrollment Manager at **530-255-2018** immediately.

9. What Happens When You Fail

- **Failing Is Growth**: This is a very challenging course, and there is a high likelihood **you will fail at least once**. Failure does not mean the same thing in this environment as it does everywhere else. In the Bethel Tech environment, failure alerts you to areas **where you need to grow** in order to succeed. It allows you to learn from your mistakes, grow where you're weak, gain more information you might have missed, and make adjustments. Learning is why you are here! **The only way to truly fail is to give up** and not try something new. So, learn from your mistakes, adjust your priorities, and try something you haven't before!
- **Extensions**: When you aren't able to complete a module by the final Saturday, you can **request an extension** 48 hours prior to the final

Saturday. Your Instructional Pastor posts a link in the #announcements channel on Slack where you will submit your request, and he will contact you with approvals.

- **3 Retakes**: Our academic policy is that you have **3 chances per module** to pass a module. You need to pass it on your 3rd retake in order to stay in the program.
- **First Retake**: Upon your **first retake**, you will be placed in **Warning** status. You are not in trouble, but this alerts our staff to keep an eye on your progress and contact you to chat about how we can better support you.
- **Second Retake**: Upon your **second retake**, you will be placed in a **Probation** status. This is your **final chance** to successfully complete your module. We will seek to discover if there are any adverse experiences occurring in your life, or if you require additional mentor or peer support. We are not here to shame or punish you, we only want to help you be as successful as we know you can be.
- **Suspension**: If you **fail a third time**, you will then be placed on **Academic Suspension** as soon as your grade is processed in our system. This is because you have made insufficient academic progress, and you will be removed from all modules and must wait 6 weeks to submit an appeal to return.
- **Emergencies:** If at any point you are experiencing circumstances that prevent you from moving forward in your module, **please be proactive and immediately contact** your Retention Specialist or Enrollment Manager.

10. Be Curious

- You are here to learn! **Learning is hard**. It does not always feel comfortable or like you expected. You are **building new neural pathways in your brain**, and this takes time.
- **Don’t be afraid to explore**. Try new things! This means changing your code, even if you’re scared you’ll break it.
- **Associate new information with what you already know**. One example that our coding mentor, Will Harris, used to help him understand how a Javascript function acts when he was a student, he associated it with a lightbulb and a lightswitch. A lightbulb has everything it needs to work, but won't without a lightswitch to turn it on or off. A function works in a similar way, and this helped him **relate a real-world example to a brand-new concept.**

- When you plant something new, **fruit is not overnight**. Take time to see the changes. Be patient with yourself, and take a moment to celebrate the little successes!
- If you're **worried about becoming a "developer"** by the end of this course, read this awesome article written by someone 6 months in. They unpack what you'll experience and tips to help you succeed. You're gonna be great :)
    - "**Ten Things I've Learned in Six Months of Being a Junior Frontend Developer**" https://dev.to/ellen_dev/ten-things-i-ve-learned-in-six-months-of-being-a-junior-frontend-developer-14n0