Talent Acceleration Program

WORKBOOK

Week 2

TAP Week 2

“Time is the scarcest resource; and unless it is managed, nothing else can be managed.”

Welcome to week 2 of the Talent Acceleration Program!

It is divided into several sections:

1. Technical Skills
2. Social Skills
3. Self Assessment 2
4. Deliverables

Good luck and enjoy this week’s content!

Part 1: Technical Skills

Week	Topics
1	Frontend Fundamentals
2	Backend Fundamentals
3	Agile Fundamentals
4	Project Management Fundamentals

Having strong technical fundamentals is an absolute must for you to qualify for a role as a professional web developer. As such, the first month of the program you'll work to refresh your understanding.

In week 2 you'll refresh your **Backend fundamentals**.

Backend Fundamentals

In any web application, the part that a user can interact with is called the frontend (also known as the user interface). These could be elements like buttons, dropdown menus or videos.

Everything else that's needed to make it actually work the way it does, is called the backend. But have you ever wondered why there is this distinction?

SEPARATION OF CONCERNS

In software design (that is, how we think we should build any particular software before we actually build it) we always want to do what's most effective: how to write code that's reusable, easy to change and modular.

Why? Because it’s the most cost-effective way to create software.

THE BUSINESS SIDE OF SOFTWARE DEVELOPMENT

Imagine a customer comes along and offers to pay you for an enhancement to their software: a user login system. In order to get paid, you will need to change your program to add the enhancement. But what constitutes how much money you can ask?

- How much code you have to change
- How easy it is to make the changes
- How likely you are to break existing features that are being used by other customers
- How much you can reuse the existing model/architecture

From a business perspective, all of these points should result in a time estimation: how long will it take to deliver upon the enhancement?

A fundamental principle that allows us to design and build cost-effective software is called separation of concerns.

In simple terms this means: each part of the application should only have 1 job.

- The 1 job of the frontend is to be an interface for the user to interact with. (i.e. easy to understand, good user experience of the company’s

product)
- The 1 job of the backend is to manipulate data in whatever way is necessary (i.e. store/retrieve data from the database, connect with APIs)

By having the job (or “concern”) clearly defined for each part, it’s easier to think about, design and implement the enhancement.

Let’s take a look again at the 4 points that affect the budget of a project, and see how this relates to the concept of separation of concerns:

- **How much code you have to change**
  If all of the code for a particular behaviour of the application is separated out, then you will only have to change code directly associated with your new feature. Which should be less code to change.

- **How easy it is to make the changes**
  If the behaviours you are interested in are neatly separated from the rest of the application it is more likely you will be able to swap in a new implementation without having to fully understand or manipulate the rest of the program. It should also be easier to find out which code you need to change.

- **How likely you are to break existing features that are being used by other customers**
  Code that you do not have to change is less likely to break than code that you do change. So splitting up the concerns helps you to avoid breakage in unrelated features by preventing you from having to change code that they could call. If your features are mixed up together

you might change the behavior of one by accident while trying to change another one.

- **How much you can reuse the existing model/architecture**
  If your architecture is agnostic to technical or business logic detail then changes to implementation are less likely to require new architectural features. For example, if your main web server logic is database agnostic then supporting a new database should be as easy as swapping in a new implementation of the persistence layer.

Learning Materials

- What is Separation of Concerns in Software Architecture

WHAT IS BACKEND

Now that we know more about the distinction between frontend and backend, let's dive deeper into the concept of backend itself.

In web development the term backend can be boiled down to 3 components:

- **A server (hardware)**: a computer that is connected to other computers, which runs an application (see below) that allows for sharing and managing services (like a calculator or word processor) and resources (like images or text files).
- **A database (software)**: manages and saves sensitive data in a structured way (i.e. lists of names, email address, etc.).

- **An application (software)**: contains code that allows it to interact with and manipulate the server (i.e. the filesystem), database and other applications.

In web development, the application part of the backend is often referred to as a web server: it *serves* resources over the Internet to some client, like the browser.

Learning Materials

- A Beginners Guide to Backend Development

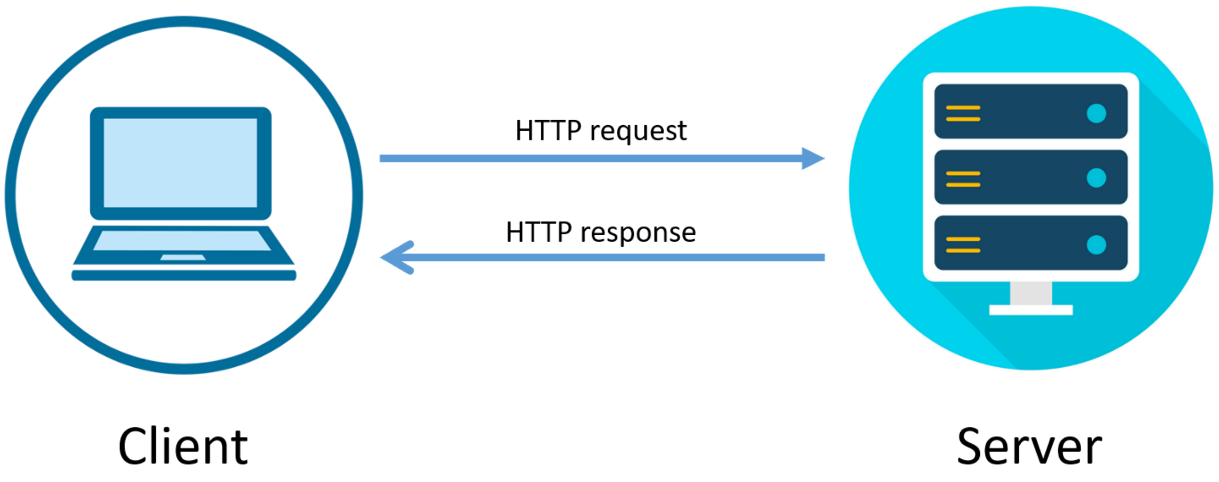
WEB SERVERS

A web server (usually shortened to "server") is an application that runs on a computer. It contains code that instructs the computer to receive a request and handle it accordingly. The request comes from a client (usually a browser, but this could also be another application), usually to retrieve or store data.

The web server interprets the request and performs what is asked for. Whether or not it has been successful, the web server always sends back a response to the client.

In the case that things didn't go as expected, the response usually contains an error message (also known as an exception). If things did succeed it sends back as a response what was asked for.

This request-response between a client and server is referred to as the client-server model.



Learning Materials

- Client-Server Model
- What Is a Web Server?
- Web Application Architecture

USING JAVASCRIPT IN THE BACKEND

Nowadays there's a way to create backend applications with JavaScript, called Node.js.

Through Node's internal package manager, NPM, we can access many different modules that can help us perform essential functions on the operating system of your computer.

Learning Materials

- Software Download
- Learn Node.js in 1 Hour

With Node.js, we can also make web servers. There are two ways of doing so: using native modules (such as *http*) or a web framework.

Whichever one you choose depends on your business needs: is it an internal project or a client project? How tailor-made should it be? What's the budget and timeline?

EXPRESS.JS

In software development we're always striving to make our jobs as easy as possible, while still keeping things customizable when needed. A web framework helps a lot with this.

A popular web framework for Node.js is Express.js. It contains all the essential features needed to rapidly build a web server, but still gives us the freedom to add any other functionality (i.e. by adding native modules included in Node.js, or any module found in the NPM ecosystem) easily.

Learning Materials

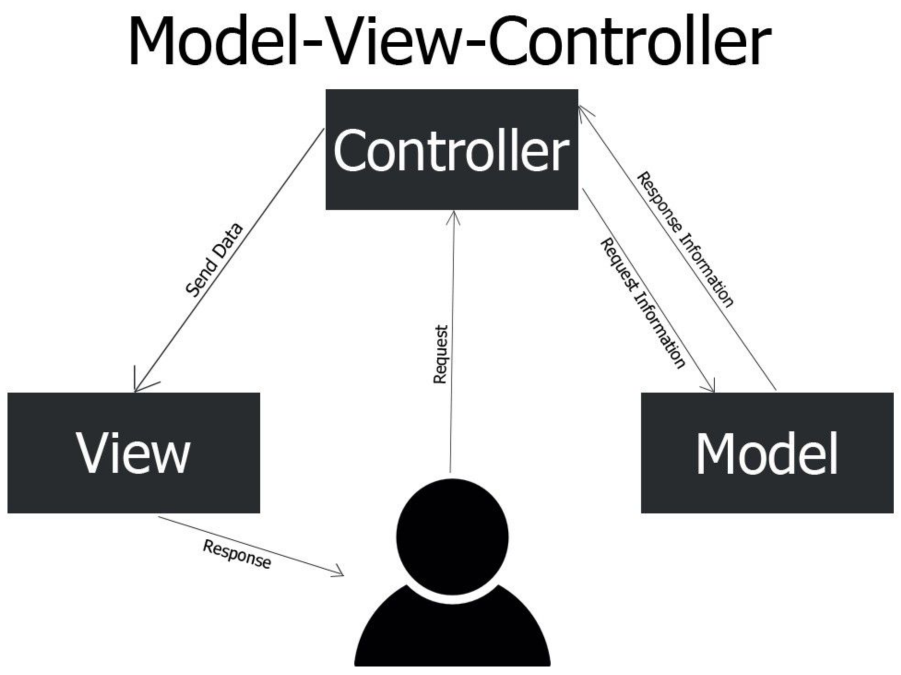
- Express.js

ARCHITECTURAL PATTERNS: MODEL-VIEW-CONTROLLER

Once you've decided to build a web server, it's important to decide *how* the features are going to relate to each other. Do you put everything into a single file? Or does it make more sense to group features according to functionality?

This is where an architectural pattern comes into play. That's a fancy way of saying: organizing your functions and variables in a robust and readable way, using various files and folders.

A common architectural pattern is called the model-view-controller (MVC).



The model represents your data definitions or schemas; the blueprint of any particular data entity in your application. A common example of this is a User or Product model.

The view represents the user interface or frontend of your application. It's the place where the data will be shown in a user-friendly way.

The controller (also known as a request handler) represents the part of the web server that moves data from one place to another. It's essentially a function that receives a request and proceeds to "handle" it, depending on what is required (i.e. store/retrieve data from the database, validate inputs from the view, etc.).

Learning Materials

- Separation of Concerns: MVC
- The Big Picture of JavaScript REST APIs

API TESTING

A big part of backend development is API testing. You always want to make sure that your backend logic works as intended.

Normally speaking, you'd use the browser or terminal for this. Send a request to your running web server and you should be able to see if it works.

However, this is not efficient as the browser might. Postman is a graphical user interface that's designed to solve one problem: make API testing as easy as possible.

Learning Materials

- How to use Postman

Technical Assignment

In order to give you a soft landing into TAP, you are not required to work on a client project just yet. Instead, you are tasked to make an assignment to solidify what you have learned. Take a look at the following:

ASSIGNMENT: The Pokédex App (backend)

Description

Using Express.js, create a web server that will (1) serve the frontend from last week and (2) provides data in JSON format.

Feature list

- Serve frontend (from last week)
- Host on Heroku

Acceptance criteria

- Uses Express.js
- Incorporates MVC pattern
- Makes use of the following static data set

How to submit

Submit your application to the TAP-Cohort-1 repository, on BitBucket. Make sure to:

- Create a branch (i.e. “noer-week2-coursework”)

- Make a pull request to “master”
- Add your mentor (in this case Noer) as a reviewer

The deadline is **Saturday February 27, 23.59**.

PART 2: Soft skills

Week	Topic
1	Growth mindset
2	Time management
3	Professional communication I
4	Customer centricity I

Where technical or hard skills are the basic minimum requirements to do a job, the development of your soft skills will be what defines your career success.

Foundations Module: Week 2

Have you ever double booked an appointment? Or arrived late at a meeting? If so, you have mismanaged your time.

But why is this so important, and what is time management anyway?

TIME IS THE MOST PRECIOUS RESOURCE YOU HAVE

We live in a remarkable time. Today we have access to an infinite wealth of information right at our fingertips, all thanks to the internet and smartphones. And the best thing is: it’s all for free. Or is it?

When you were born you were given a set amount of the most precious resource there is. It’s not your brains, nor your family; it’s time.

Every moment you're engaging things that will not improve the quality of your life, like mindlessly browsing social media, you are actually paying for it. But it's not money you spend, but time!

And that's one resource you can never get more of; once it's used it's gone forever.

Learning Materials

- This is how short life is
- How to Master Your Most Valuable Resource

PRODUCTIVITY

In business we strive to use our time as productively as possible. Let's imagine you are running a lemonade stand. Your business goal is to sell as much lemonade as possible.

In order to start selling, you need to produce lemonade. You first need the get the required ingredients. Then you need to spend time actually making it. If you're a beginner you're probably inconsistent, so the quality might not always be the best. So sometimes it takes an hour to make a good batch, other times you have it right on the first try.

Your ability to be more consistent in the quality you deliver is what we could call "being productive".

A simple definition of productivity, therefore, could be:

the amount of business value that can be created,
in as short a time as possible.

And what's that business value? Anything that the business could use to realize their goals; whether that's making profit, social impact or just plain fun.

In the case of your lemonade stand, examples of business value are (1) the quality of your ingredients, (2) your production method and (3) selling skills. The better those are, the more productive your business is!

Learning Materials

- What Is Productivity?
- How To Be 10X More Productive

PROCRASTINATION

All this might sound quite nice, but you're not a machine. Sometimes you don't 'feel like it' and thus procrastinate.

In your private life you are free to do so if you don't feel like it. You don't *have* to exercise every week, right?

Things are different in business: maybe you need to make a deadline, or your colleagues are waiting for you to finish so they can move forward. You also have much more at stake: your financial security.

Regardless of the situation, there are many reasons for putting off on what you're supposed to do. Not knowing what to do, perfectionism, or just plain laziness. What are the sources of your procrastination?

Learning Materials

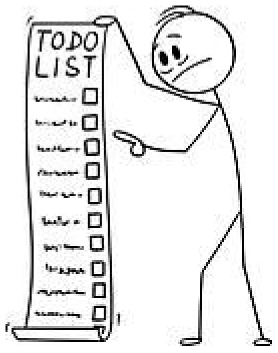
- Why do We Procrastinate?
- Procrastination

TIME MANAGEMENT VS. ENERGY MANAGEMENT

So time management is important, because our lives are short and we want to make sure we spend our time (aka our life) on the things that matter.

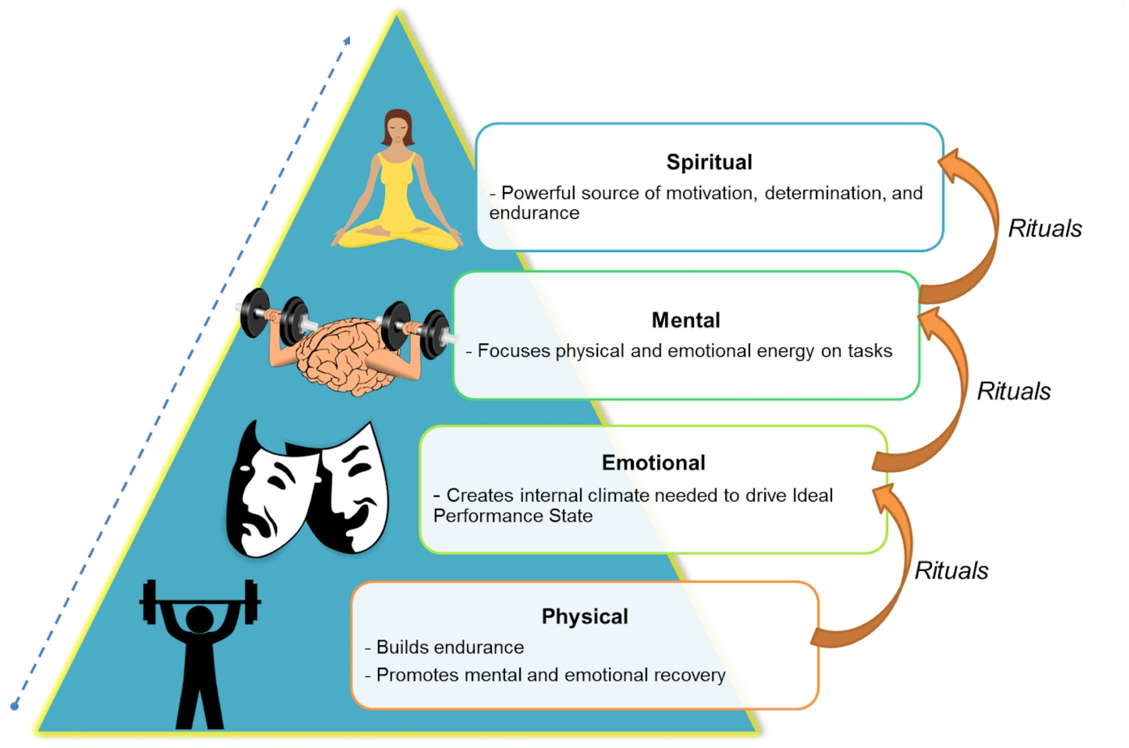
This is nice and all, but we all have moments where it's just too hard to do the right thing in order to be effective.

Let's imagine you have a long to-do-list, but you just can't get to it because *you're tired*. Is time management really what you should be concerned with?



If you think about it, it's actually energy that allows you to do tasks in the first place. Of course, it's still important to keep track of time and make sure you plan effectively. But managing yourself should always be the foundation.

Like anything, there are several theories on this. One worth looking into is the High-Performance Pyramid by Jim Loehr and Tony Schwartz.


Spiritual
- Powerful source of motivation, determination, and endurance
Mental
- Focuses physical and emotional energy on tasks
Emotional
- Creates internal climate needed to drive Ideal Performance State
Physical
- Builds endurance
- Promotes mental and emotional recovery
Rituals
Rituals
Rituals

Looking at the image above, you see 4 sources of energy. As software developers you can imagine that we use mental energy most of the time. Writing code is hard work and the brain burns a lot of calories doing it!

Since it's energy we're talking about, is it weird to assume that it will run out at some point? And if it does, the logical next step is to take a rest.

Learning Materials

- Making Time Management Work for You
- The Pomodoro Technique
- Manage Your Energy, Not Your Time
- The Power of Full Engagement

Tools

- Tomighty (Pomodoro Timer)
- Google Calendar for Slack

Note: In order to make these abstract soft skills topics more concrete, there will be a Group Exercise done every Thursday. Together with the rest of the cohort we'll bring the concepts to life!

Self-Assessment Form

In this form you will find various questions that will challenge you to reflect on your learning process. Learning new skills takes time, and a good way to make the most out of it is to think actively about what you're doing and why.

How to answer the questions:

BAD EXAMPLE

- *This week I learned about Conflict Resolution. It's difficult.*

GOOD EXAMPLE

- *This week I learned about Conflict Resolution, which was easy for me to understand. It's an important concept, because it helps mitigate tension in our (business) relationships. An example of this is one argument I had with a colleague. We disagreed on something, but I could quickly recognize that it was just a simple misunderstanding.*

1. Technical Skills
    a. What are 2 essential functions of a web server?
    b. What are 2 benefits of using Postman?
2. Soft Skills
    a. How can this week's concept (*Time management*) help you be a better professional?
    b. What's your main source of procrastination and what's 1 solution for it?
    c. Why is energy management just as important as time management?
3. Business Awareness
    a. From a business perspective, why should we write reusable code?
    b. Why is time estimation for software development so important?

Deliverables

Each week you have to deliver on what you've learned.

The deliverables for this week are:

1. Technical assignment: the Pokédex App
2. Self-assessment Form 2

You are expected to deliver all files in a pull request to the master branch of the tap-cohort-1 repository on BitBucket.

The **deadline** is **February 27, 23.59**.