

Get a free sample

# The Lessons

A WHOLE LOT OF THEM!

## Section 1: Getting Started

- ▶ A Basic Web Application (Sample)
- ▶ Troubleshooting and Slack
- ▶ Packages and Imports (Sample)
- ▶ Editors and Automatic Imports (Sample)
- ▶ The "Hello, World" Part of our Code (Sample)
- ▶ Web Requests (Sample)
- ▶ HTTP Methods (Sample)
- ▶ Our Handler Function (Sample)
- ▶ Registering our Handler Function... (Sample)
- ▶ Go Modules (Sample)

## Section 2: Adding New Pages

- ▶ Dynamic Reloading (Sample)
- ▶ Setting Header Values (Sample)
- ▶ Creating a Contact Page (Sample)
- ▶ Examining the `http.Request` Type (Sample)
- ▶ Custom Routing (Sample)
- ▶ URL Path vs `RawPath`
- ▶ Not Found Page
- ▶ The `http.Handler` Type

- ▶ The `http.HandlerFunc` Type
- ▶ Exploring Handler Conversions
- ▶ FAQ Exercises

## Section 3: Routers and 3rd Party Libraries

- ▶ Defining our Routing Needs
- ▶ Using `git`
- ▶ Installing `Chi`
- ▶ Using `Chi`
- ▶ `Chi` Exercises

## Section 4: Templates

- ▶ What are Templates?
- ▶ Why Do We Use Server Side Rendering?
- ▶ Creating Our First Template
- ▶ Cross Site Scripting (XSS)
- ▶ Alternative Template Libraries
- ▶ Contextual Encoding
- ▶ Home Page via Template
- ▶ Contact Page via Template
- ▶ FAQ Page via Template
- ▶ Template Exercises

## Section 5: Code Organization

- ▶ Code Organization
- ▶ MVC Overview
- ▶ Walking Through a Web Request with MVC
- ▶ MVC Exercises

## Section 6: Starting to Apply MVC

- ▶ Creating the Views Package
- ▶ `fmt.Errorf`
- ▶ Validating Templates at Startup
- ▶ Must Functions
- ▶ Exercises

## Section 7: Enhancing our Views

- ▶ Embedding Template Files
- ▶ Variadic Parameters
- ▶ Named Templates
- ▶ Dynamic FAQ Page
- ▶ Reusable Layouts
- ▶ Tailwind CSS
- ▶ Utility-first CSS
- ▶ Adding a Navigation Bar
- ▶ Exercises

## Section 8: The Signup Page

- ▶ Creating the Signup Page
- ▶ Styling the Signup Page
- ▶ Intro to REST
- ▶ Users Controller
- ▶ Decouple with Interfaces
- ▶ Parsing the Signup Form
- ▶ URL Query Params
- ▶ Exercises

## Section 9: Databases and PostgreSQL

- ▶ Intro to Databases

- ④ Intalling Postgres
- ④ Connecting to Postgres
- ④ Update: Docker Container Names
- ④ Creating SQL Tables
- ④ Postgres Data Types
- ④ Postgres Constraints
- ④ Creating a Users Table
- ④ Inserting Records
- ④ Querying Records
- ④ Filtering Queries
- ④ Updating Records
- ④ Deleting Records
- ④ Additional SQL Resources

## Section 10: Using Postgres with Go

- ④ Connecting to Postgres with Go
- ④ Imports with Side Effects
- ④ Postgres Config Type
- ④ Executing SQL with Go
- ④ Inserting Records with Go
- ④ SQL Injection
- ④ Acquire a new Record's ID
- ④ Querying a Single Record
- ④ Creating Sample Orders
- ④ Querying Multiple Records
- ④ ORMs vs SQL
- ④ Exercises
- ④ Syncing the Book and Screencasts Source Code

## Section 11: Securing Passwords

- ▶ Steps for Securing Passwords
- ▶ Third Party Authentication Options
- ▶ What is a Hash Function?
- ▶ Store Password Hashes, Not Encrypted or Plaintext Values
- ▶ Salt Passwords
- ▶ Learning bcrypt with a CLI
- ▶ Hashing Passwords with bcrypt
- ▶ Comparing a Password with a bcrypt Hash

## Section 12: Adding Users to our App

- ▶ Defining the User Model
- ▶ Creating the UserService
- ▶ Create User Method
- ▶ Postgres Config for the Models Package
- ▶ UserService in the Users Controller
- ▶ Create Users on Signup
- ▶ Sign In View
- ▶ Authenticate Users
- ▶ Process Sign In Attempts

## Section 13: Remembering Users with Cookies

- ▶ Stateless Servers
- ▶ Creating Cookies
- ▶ Viewing Cookies with Chrome
- ▶ Viewing Cookies with Go
- ▶ Securing Cookies from XSS
- ▶ Cookie Theft
- ▶ CSRF Attacks
- ▶ CSRF Middleware

- ④ Providing CSRF to Templates via Data
- ④ Custom Template Functions
- ④ Adding the HTTP Request to Execute
- ④ Request Specific CSRF Template Function
- ④ Template Function Errors
- ④ Securing Cookies from Tampering
- ④ Cookie Exercises

## Section 14: Sessions

- ④ Random Strings with `crypto/rand`
- ④ Exploring `math/rand`
- ④ Wrapping the `crypto/rand` package
- ④ Why Do We Use 32 Bytes for Session Tokens?
- ④ Defining the Sessions Table
- ④ Stubbing the `SessionService`
- ④ Sessions in the Users Controller
- ④ Cookie Helper Functions
- ④ Create Session Tokens
- ④ Refactor the `rand` Package
- ④ Hash Session Tokens
- ④ Insert Sessions into the Database
- ④ Update Existing Sessions
- ④ Query Users via Session Token
- ④ Deleting Session
- ④ Sign Out Handler
- ④ Sign Out Link
- ④ Session Exercises

## Section 15: Improved SQL

- ④ SQL Relationships

- ▶ Foreign Keys
- ▶ On Delete Cascade
- ▶ Inner Join
- ▶ Left, Right, and Full Outer Join
- ▶ Using Join in the SessionService
- ▶ SQL Indexes
- ▶ Creating PostgreSQL Indexes
- ▶ On Conflict
- ▶ Improved SQL Exercises

## Section 16: Schema Migrations

- ▶ What are Schema Migrations?
- ▶ How Migration Tools Work
- ▶ Installing pressly/goose
- ▶ Converting to Schema Migrations
- ▶ Schema Versioning Problem
- ▶ Run Goose with Go
- ▶ Embedding Migrations
- ▶ Go Migration Files
- ▶ Removing Old SQL Files

## Section 17: Current User via Context

- ▶ Using Context to Store Values
- ▶ Improved Context Keys
- ▶ Context Values with Types
- ▶ Storing Users as Context Values
- ▶ Reading Request Context Values
- ▶ Set the User via Middleware
- ▶ Requiring a User via Middleware

- ▶ Accessing the Current User in Templates
- ▶ Request-Scoped Values

## Section 18: Sending Emails to Users

- ▶ Password Reset Overview
- ▶ SMTP Services
- ▶ Building Emails with SMTP
- ▶ Sending Emails with SMTP
- ▶ Building an Email Service
- ▶ EmailService.Send
- ▶ Forgot Password Email
- ▶ ENV Variables

## Section 19: Completing the Authentication System

- ▶ Password Reset DB Migration
- ▶ Password Reset Service Stubs
- ▶ Forgot Password HTTP Handler
- ▶ Asynchronous Emails
- ▶ Forgot Password HTML Template
- ▶ Initializing Services with ENV Vars
- ▶ Check Your Email HTML Template
- ▶ Reset Password HTTP Handlers
- ▶ Reset Password HTML Template
- ▶ Update Password Function
- ▶ Implementing PasswordReset.Create
- ▶ Implementing PasswordReset.Consume

## Section 20: Better Errors

- ▶ Inspecting Errors
- ▶ Inspecting Wrapped Errors



- 🔍 Designing the Alert Banner
- 🔍 Dynamic Alerts
- 🔍 Removing Alerts with JavaScript
- 🔍 Detecting Existing Emails
- 🔍 Accepting Errors in Templates
- 🔍 Public vs Internal Errors
- 🔍 Creating Public Errors
- 🔍 Using Public Errors
- 🔍 Better Error Handling Exercises

## Section 21: Galleries

- 🔍 Galleries Overview
- 🔍 Gallery Model and Migration
- 🔍 Creating Gallery Records
- 🔍 Querying for Galleries by ID
- 🔍 Querying Galleries by UserID
- 🔍 Updating Gallery Records
- 🔍 Deleting Gallery Records
- 🔍 New Gallery Handler
- 🔍 `views.Template` Name Bug
- 🔍 New Gallery Template
- 🔍 Gallery Routing and CSRF Bug Fixes
- 🔍 Create Gallery Handler
- 🔍 Edit Gallery Handler
- 🔍 Edit Gallery Template`
- 🔍 Update Gallery Handler
- 🔍 Gallery Index Handler
- 🔍 Discovering and Fixing a Gallery Index Bug
- 🔍 Gallery Index Template Continued

- ④ Show Gallery Handler
- ④ Show Gallery Template and a Tailwind Update
- ④ Extracting Common Gallery Code
- ④ Extra Gallery Checks with Functional Options
- ④ Delete Gallery Handler
- ④ Gallery Exercises

## Section 22: Images

- ④ Images Overview
- ④ Setting Up Test Images
- ④ Adding the ImagesDir to the GalleryService
- ④ Globbing Image Files
- ④ Adding Filename and GalleryID to the Image Type
- ④ Adding Images to the Show Gallery Page
- ④ Show Image Handler
- ④ Querying for a Single Image
- ④ URL Path Escaping Image Filenames
- ④ Adding Images to the Edit Gallery Page
- ④ Delete Image Form
- ④ Delete Image Service Func
- ④ Delete Image Handler
- ④ Checking for Filename Vulnerabilities
- ④ Upload Image Form
- ④ Image Upload Handler
- ④ Creating Images in the GalleryService
- ④ Detecting Content Type
- ④ Rendering Content Type Errors

## Section 23: Preparing for Production

- ④ Loading All Config via ENV
- ④ Docker Compose Overrides
- ④ Building Tailwind Locally
- ④ Tailwind Via Docker
- ④ Serving Static Assets
- ④ Making main Easier to Test
- ④ Running our Go Server via Docker
- ④ Multi-Stage Docker Builds
- ④ Tailwind Production Build
- ④ Caddy Server via Docker

## Section 24: Deploying

- ④ Creating a Digital Ocean Droplet
- ④ Setting up DNS
- ④ Installing Git on the Server
- ④ Setting Up a Bare Git Repo
- ④ Setting Up a Local Git Repo
- ④ Checking Out Our Code on the Server
- ④ Email Sending Server Setup
- ④ Production .env File
- ④ Install Docker in Prod
- ④ Production Caddyfile
- ④ Production Data Directories
- ④ Running Our App in Prod
- ④ Post-receive Deploy Updates
- ④ Deploy via Git
- ④ Logging Services

## Bonus: OAuth

- ▶ Intro to OAuth
- ▶ OAuth Example Code
- ▶ Dropbox App Setup
- ▶ Offline OAuth Demo
- ▶ OAuth Tokens
- ▶ Online vs Offline Access Types
- ▶ Redirect URIs
- ▶ OAuth Connect HTTP Handler
- ▶ Determine Redirect URI Host
- ▶ OAuth Routes and Config Setup
- ▶ OAuth Callback Handler
- ▶ Testing OAuth with API Calls

## Bonus: Dropbox Chooser

- ▶ Dropbox Chooser Overview
- ▶ Embedding the Chooser
- ▶ Images via Dropbox Form
- ▶ Chooser Success Function
- ▶ Images via URL Handler
- ▶ Downloading Images
- ▶ Create Images Without Seek
- ▶ Concurrent Downloads
- ▶ Using errgroup
- ▶ Page Specific JS