**Personal Competence Plan**

# Introduction

**What Do I Want to Learn?**

## Knowledge:

### Advanced Go programming patterns.

### In-depth understanding of database systems and ORM design.

### Best practices in software architecture and performance optimization.

## Skills:

### Designing and developing a robust, high-performance library in Go.

### API design and documentation.

### Testing, debugging, and code optimization.

### Version control and collaboration (using Git, GitHub, etc.).

**Why Do I Want to Learn This?**

## **Personal Improvement**: *I want to become better at building efficient, maintainable software* because it enables me to create tools that solve real-world problems and contribute to the community.

## **Professional Development**: Enhancing my skills in Go and ORM design will increase my value in the job market and help me pursue a career in software engineering.

## **Project Motivation**: My dissatisfaction with existing solutions (like GORM) drives me to design an improved, more efficient Go ORM library that reflects my own standards for simplicity and performance.

**What Is My Starting Level?**

## Experience:

### Solid foundation in Go programming.

### Basic knowledge of ORM concepts and some exposure to GORM.

### Familiarity with software development practices (version control, testing, etc.).

## Realism & Ambition:

### **Realistic**: I have the background to build on, with enough familiarity to learn advanced concepts.

### **Ambitious**: Developing an entire ORM library is a challenging goal that pushes me to acquire new, in-depth skills.

# How

**How Do I Like to Accomplish My Goals?**

## Learning Methods:

### **Self-Study**: Online courses, tutorials, and documentation (e.g., Go’s official docs, community guides).

### **Workshops & Seminars**: Attend sessions focusing on advanced Go programming, database systems, and software design.

### **Hands-On Coding**: Engage in coding assignments and build a personal project (the Go ORM library).

### **Peer & Expert Feedback**: Regularly share progress with coaches, fellow students, and experts (internal Fontys ICT mentors).

**What Are the Relevant Learning Goals?**

## Master advanced Go language constructs and idioms.

## Understand and implement effective ORM patterns.

## Develop skills in API design and library architecture.

## Learn best practices in testing, documentation, and performance benchmarking.

**What Kind of Subjects and Activities Must I Carry Out?**

## Subjects:

### Advanced Go programming.

### Database design and management.

### Software architecture and design patterns.

### Performance optimization and testing strategies.

## Activities:

### Completing relevant tutorials and online courses.

### Participating in coding workshops and seminars.

### Engaging in regular coding assignments and building the personal project.

### Setting up code review sessions and seeking expert feedback.

### Attending peer group meetings and participating in online forums.

# Project

**How Does a Project Fit My Learning Goals?**

## Group Project (Optional):

### Collaborate on an open-source project integrating advanced database and software architecture topics.

### Learn teamwork, communication, and collaborative problem-solving.

## Personal Project:

### **Project Title**: TORM (working title)

### **Objective**: Develop a lightweight, efficient ORM library in Go.

### **Relevance**: This project directly addresses my goal of creating a better alternative to GORM by applying advanced programming and design concepts.

### **Activities Involved**:

#### Designing the library architecture.

#### Coding core features (e.g., CRUD operations, query building, transaction management).

#### Writing unit and integration tests.

#### Documenting the API and gathering iterative feedback from peers and experts.

# Proof

**How Will I Check and Prove I Have Acquired Knowledge/Skills?**

## Testing and Demos:

### Create Proof-of-Concept (PoC) implementations of key features.

### Build a demo that showcases a complete workflow (connecting to a database, performing CRUD operations, etc.).

## Visibility:

### Publish the project on GitHub with detailed documentation.

### Regularly update a progress log or blog.

### Participate in code reviews and present demos to peers and experts.

## Feedback:

### Solicit expert feedback and engage in structured peer review sessions.

### Use performance benchmarks and tests to validate improvements.

# Validate

**How Do I Validate My Subjects, Activities, and Goals?**

## Online Resources:

### Use reputable online learning platforms (e.g., Udemy, Coursera, Pluralsight) and verify resources with expert reviews.

## Expert Involvement:

### Leverage expertise available at Fontys ICT and from the broader Go community.

### Engage coaches and fellow students in regular feedback loops.

## External Validation:

### Take quizzes, complete assignments, and participate in coding challenges.

### Compare project progress and performance with industry benchmarks.

## Support Network:

### Utilize mentors and peer groups for ongoing validation and guidance.

# Planning

**What Challenges or Problems Can I Expect?**

## Challenges:

### Integrating new design patterns with Go’s native practices.

### Ensuring high performance while maintaining simplicity.

### Overcoming potential design conflicts or performance bottlenecks.

## Strategies:

### Regularly schedule review sessions and adjust the plan based on feedback.

### Research and experiment with different approaches before finalizing decisions.

### Use agile methods to iterate quickly and pivot if necessary.

**What Do I Need to Learn This Properly?**

## Access to online courses, expert mentors, and updated documentation.

## Practical tools (IDE, version control, CI/CD setup) and collaboration platforms.

## A supportive network of peers and industry professionals for feedback and advice.

**What Will Be the Milestones for This Semester?**

## Initial Setup & Research:

### Define the project scope and set up the Git repository.

### Complete foundational tutorials in advanced Go and ORM concepts.

## Design & Planning Phase:

### Create architectural sketches, wireframes, and initial API designs.

### Gather initial feedback from peers and experts.

## Development Phase:

### Implement core functionalities (database connections, CRUD operations, basic query builder).

### Integrate testing frameworks and document progress.

## Mid-Semester Review:

### Present a working PoC to coaches and peers.

### Evaluate progress and adjust learning goals if necessary.

## Finalization & Demo:

### Refine features based on feedback and conduct performance benchmarking.

### Complete documentation and host a final demo session.

### Make the project publicly available on GitHub.