ການສຶກສາປຽບທຽບປະສິດທິພາບການປະມວນຜົນຂອງພາສາໂປຣແກຣມ GO ແລະ ພາສາໂປຣແກຣມ TypeScript ໃນການພັດທະນາ Web Application

ຄອນສະຫວັນ ຂຸນວິເສດ[*], *ຄຳພັດ ບຸນນະດີ*[2], *ທາ ບຸນທັນ*[3], *ວິມົນທາ ແກ້ວວົງພະຈັນ*[4], *ພອນໄຊ ວິລະກອນ*[5]

ພາກວິຊາວິສະວະກຳຄອມພິວເຕີ ແລະ ເຕັກໂນໂລຢີຂໍ້ມູນຂ່າວສານ, ຄະນະວິສະວະກຳສາດ, ມະຫາວິທະຍາໄລແຫ່ງຊາດລາວ, ສປປ ລາວ

ບົດຄັດຫຍໍ້

Web Application ມີບົດບາດສຳຄັນຢ່າງຍິ່ງໃນຊີວິດປະຈຳວັນ ແລະ ການເຮັດທຸລະກິດ, ເຊິ່ງໄດ້ກາຍເປັນສ່ວນໜຶ່ງທີ່ຕັດແຍກບໍ່ໄດ້. ການພັດທະນາ Web Application ສາມາດນຳໃຊ້ພາສາໂປຣແກຣມທີ່ຫຼາກຫຼາຍ, ໂດຍສະເພາະ Go ແລະ Typescript ເຊິ່ງເປັນພາສາໃໝ່ທີ່ໄດ້ຮັບຄວາມນິຍົມສູງຍ້ອນຄຸນສົມບັດທີ່ທັນສະໄໝ. ການສຶກສາຄົ້ນຄວ້ານີ້ມີຈຸດປະສົງເພື່ອປຽບທຽບຄວາມໄວໃນການຕອບສະໜອງຂອງ Web Application ທີ່ພັດທະນາດ້ວຍສອງພາສາດັ່ງກ່າວ ໂດຍສະເພາະ ເພື່ອປຽບທຽບຄວາມໄວໃນການຕອບສະໜອງຂອງ Web Application ທີ່ສ້າງດ້ວຍພາສາ Go ແລະ Typescript ໃນການຈັດການກັບຖານຂໍ້ມູນ (Select, Insert, Update, Delete). ການທົດລອງໄດ້ດຳເນີນການໂດຍການຕິດຕັ້ງ Web Application ທີ່ພັດທະນາດ້ວຍ Go ແລະ Typescript ໄວ້ທີ່ເຄື່ອງແມ່ຂ່າຍ (server). ຈາກນັ້ນ, ເຄື່ອງລູກຂ່າຍ (client) ຈະດຳເນີນການເກັບກຳຂໍ້ມູນປະສິດທິພາບໃນການຈັດການຖານຂໍ້ມູນໂດຍໃຊ້ຄຳສັ່ງ Select, Insert, Update, ແລະ Delete ກັບຈຳນວນຂໍ້ມູນຕັ້ງແຕ່ 10 ຫາ 100,000 ແຖວ. ຜ່ານການທົດລອງ, ພົບວ່າ Web Application ທີ່ສ້າງດ້ວຍພາສາ Go ມີແນວໂນ້ມທີ່ຈະມີປະສິດທິພາບການປະມວນຜົນທີ່ດີກວ່າ ແລະ ສາມາດຈັດການ Concurrency ໄດ້ງ່າຍກວ່າ Typescript ໂດຍບໍ່ຈຳເປັນຕ້ອງເພິ່ງພາ Worker Threads ທີ່ອາດຈະມີຄວາມສັບສົນຫຼາຍກວ່າ. ຜົນໄດ້ຮັບນີ້ຊີ້ໃຫ້ເຫັນວ່າ Go ມີຄວາມເໝາະສົມກວ່າສຳລັບລະບົບທີ່ຕ້ອງການປະສິດທິພາບສູງ ແລະ ການຈັດການ Concurrency ໄດ້ດີ. ຜົນການສຶກສານີ້ສາມາດນຳໃຊ້ເປັນຂໍ້ມູນປະກອບການຕັດສິນໃຈສຳລັບນັກພັດທະນາໃນການເລືອກເທັກໂນໂລຢີທີ່ເໝາະສົມກັບໂປຣເຈັກຂອງຕົນ. ສຳລັບໂປຣເຈັກທີ່ເນັ້ນປະສິດທິພາບການປະມວນຜົນ ແລະ ການຈັດການ Concurrency, Go ອາດເປັນທາງເລືອກທີ່ດີກວ່າ.

ຄຳສັບສຳຄັນ: *GO, Typescript, ປະສິດທິພາບການປະມວນຜົນ, ເວັບແອັບພລິເຄຊັນ, ການປຽບທຽບ*

**ຕິດຕໍ່ພົວພັນ: ຄອນສະຫວັນ ຂຸນວິເສດ, ເບີໂທ: 020 5693 8159, ອີເມວ: connectlao.khone@gmail.com*

A Comparative Study of the Performance of GO and TypeScript Programming Languages in Web Application Development

Khonesavanh KHOUNVISETH[*], Khampheth BOUNNADY[2], Tha BOUNTHAN[3], Vimontha KHEOVONGPHACHANH[4], Phonexay VILAKONE[5]

Department of Computer Engineering and Information Technology, Faculty of Engineering, National University of Laos, Lao PDR

Abstract

Web Applications play a very important role in daily life and business, which has become an inseparable part. Web Application development can use a variety of programming languages, especially Go and Typescript, which are new languages that have gained popularity due to their modern features. This research study aims to compare the response speed of Web Applications developed in these two languages, especially to compare the response speed of Web Applications developed in GO and Typescript in database management (Select, Insert, Update, Delete). The experiment was conducted by installing Web Applications developed in Go and Typescript on a server. Then, the client machine will collect data on database management performance using Select, Insert, Update, and Delete commands with data sizes ranging from 10 to 100,000 rows. Through the experiment, it was found that Web Applications developed in Go tend to have better processing efficiency and can handle concurrency more easily than Typescript without having to rely on Worker Threads which can be more complicated. This result indicates that Go is more suitable for systems that require high performance and good concurrency management. The results of this study can be used as decision-making information for developers to choose the right technology for their projects. For projects that focus on processing efficiency and concurrency management, Go may be a better choice.

Key words: *GO, Typescript, processing performance, web applications, performance comparison*

***Correspondence:** Khonesavanh KHOUNVISETH; Tel: 020 5693 8159; Email: connectlao.khone@gmail.com*

1. ພາກສະເໜີ

1.1 ຄວາມເປັນມາ ແລະ ຄວາມສຳຄັນຂອງບັນຫາ

ໃນປະຈຸບັນເທັກໂນໂລຢີ ແມ່ນມີການພັດທະນາໃນຫຼາຍໆດ້ານ ການໃຊ້ຊີວິດປະຈຳວັນຂອງຄົນເຮົາກໍ່ເລີ່ມຫັນມາໃຊ້ເທັກໂນໂລຢີເຂົ້າມາຊ່ວຍເພື່ອຄວາມສະດວກ, ງ່າຍ, ວ່ອງໄວ ໃນການສົ່ງຂໍ້ມູນ ແລະ ການເຂົ້າເຖິງຂໍ້ມູນຕ່າງໆ ໃນສື່ສັງຄົມອ່ອນລາຍ; ໂດຍສະເພາະ ແມ່ນການຄົ້ນຫາຂໍ້ມູນທາງອິນເຕີເນັດ ຜ່ານ ລິ້ງຂອງເວັບໄຊ ຕ່າງໆເຊັ່ນ: www.google.com, www.youtube.com ເປັນຕົ້ນ.

ການປະມວນຜົນຂໍ້ມູນ ແມ່ນຂະບວນການຄິດ ຫຼື ການຈັດລະບຽບແບບແຜນຂອງຂໍ້ມູນ ເພື່ອໃຫ້ໄດ້ຜົນຮັບຕາມທີ່ຕ້ອງການ ຊຶ່ງເຮັດໄດ້ໂດຍການຄຳນວນ, ການເຄື່ອນຍ້າຍຂໍ້ມູນ, ການປຽບທຽບ ແລະ ການວິເຄາະຂໍ້ມູນ ໂດຍອາດໃຊ້ສູດທາງຄະນິດສາດ ຫຼື ວິທະຍາສາດ, ວິທີການຕ່າງໆ ໂດຍອາໃສຄຳສັ່ງ ຫຼື ໂປຣແກຣມທີ່ຂຽນຂຶ້ນ.

ພາສາ Go ຫຼື Golang ເປັນພາສາໂປຣແກຣມມິ່ງ ທີ່ມີການພັດທະນາໂດຍ Google ແລະ ຖືກອອກແບບມາເພື່ອໃຫ້ມີປະສິດທິພາບສູງ, ມີຄວາມປອດໄພ ແລະ ເໝາະສົມກັບການພັດທະນາໂປຣແກຣມແບບ Concurrent ຫຼືການເຮັດວຽກພ້ອມກັນຫຼາຍເທຣດ (Concurrency) ຊຶ່ງເປັນສ່ວນໜຶ່ງຂອງຄວາມເປັນພາສາ ຊຶ່ງມີຄວາມເປັນເນື້ອຫາເດັ່ນຂອງ Golang; ຖືກອອກແບບຂຶ້ນມາໃນປີ 2009 ໂດຍ Rob Pike, Ken Thompson, ແລະ Robert Griesemer ຊຶ່ງເປັນຜູ້ພັດທະນາທີ່ມີຊື່ສຽງໃນໂລກໂປຣແກຣມມິ່ງ, ຄວາມໄວໃນການ compile. Go ສູງ ແລະ ຊ່ວຍໃຫ້ໂປຣແກຣມທີ່ຂຽນດ້ວຍ Go ມີປະສິດທິພາບສູງໃນການເຮັດວຽກ ໂດຍສະເພາະການເຮັດວຽກກັບ Web ແລະ Server. ພາສາ Go ເປັນພາສາແບບ compile ໄດ້ ແລະ ເປັນພາສາແບບ Statically typed ຊຶ່ງໝາຍຄວາມວ່າ ຕ້ອງປະກາດຕົວປ່ຽນກ່ອນເຮັດວຽກ ແລະ ມີເຄື່ອງມືຊ່ວຍການຈັດການຂໍ້ຜິດພາດ (Error Handling) ທີ່ດີ ໂດຍ Set ຂອງ function ແລະ ການເຮັດວຽກຂອງ code ຈະຂຽນໃນຮູບແບບຂອງ Module ທີ່ຊ່ວຍໃຫ້ການຈັດການ code ແລະ ການແບ່ງແຍກ code ມີຄວາມສະດວກສະບາຍ ແລະ ເປັນລະບຽບຫຼາຍຂຶ້ນ.

Typescript (TS) ເປັນພາສາຂຽນໂປຣແກຣມລະດັບສູງ ທີ່ບໍ່ເສຍຄ່າ ແລະ ເປັນ Opensource ທີ່ພັດທະນາໂດຍ Microsoft ທີ່ເພີ່ມການພິມແບບຄົງທີ່ດ້ວຍຄຳບັນຍາຍປະເພດທາງເລືອກໃຫ້ກັບ JavaScript ມັນໄດ້ຖືກອອກແບບສຳລັບການພັດທະນາ Application ຂະໜາດໃຫຍ່ແລະ transpiles ກັບ JavaScript; Typescript ອາດຈະຖືກນຳໃຊ້ເພື່ອພັດທະນາ Application JavaScript ສຳລັບທັງສອງ client-side ແລະ server-side ປະຕິບັດ(ເຊັ່ນດຽວກັນກັບ Node.js, Deno ຫຼື Bun). ມີຫຼາຍທາງເລືອກທີ່ມີຢູ່ສຳລັບການ transpilation ສາມາດນຳໃຊ້ Typescript Compiler ໃນຕອນຕົ້ນ ຫຼື ສາມາດເອົ້ນເອົາຕົວສັງລວມຂອງ Babel ເພື່ອປ່ຽນ Typescript ເປັນ

JavaScript. ຍັງສະໜັບສະໜຸນໄຟລ໌ຄຳນິຍາມທີ່ສາມາດບັນຈຸຂໍ້ມູນປະເພດຂອງ ຫ້ອງສະໝຸດ JavaScript ທີ່ມີຢູ່ແລ້ວ, ຄືກັນກັບໄຟລ໌ header C++ ສາມາດອະທິບາຍໂຄງສ້າງຂອງໄຟລ໌ວັດຖຸທີ່ມີຢູ່ແລ້ວ ເຮັດໃຫ້ໂຄງການອື່ນໃຊ້ຄ່າທີ່ກຳນົດໄວ້ໃນໄຟລ໌ຄືກັບວ່າ ຖືກພິມແບບສະຖິຕິ Typescript entities ມີໄຟລ໌ສ່ວນຫົວຂອງພາກສ່ວນທີ່ສາມສຳລັບ libraries ທີ່ນິຍົມເຊັ່ນ jQuery, MongoDB ແລະ D3.js. ສ່ວນຫົວ Typescript ສຳລັບໂມດູນ libraries; Node.js ຍັງມີຢູ່ ເຊິ່ງອະນຸຍາດໃຫ້ພັດທະນາໂປຣແກຣມ Node.js ພາຍໃນ Typescript.

Typescript compiler ແມ່ນຕົວຂອງມັນເອງຂຽນໃນ Typescript ແລະ ລວບລວມ ເປັນ JavaScript. ມັນໄດ້ຖືກອະນຸຍາດພາຍໃຕ້ Apache License 2.0. Anders Hejlsberg, ສະຖາປະນິກຊັ້ນນຳຂອງ C# ແລະ ຜູ້ສ້າງ Delphi ແລະ Turbo Pascal, ໄດ້ເຮັດວຽກກ່ຽວກັບການພັດທະນາ Typescript.

ສະນັ້ນ, ການເລືອກພາສາທີ່ຈະມາປະມວນຜົນຂໍ້ມູນໃນ Web application ທີ່ມີປະສິດທິພາບຈຶ່ງເປັນບັນຫາສຳຄັນສຳລັບນັກພັດທະນາໂປຣແກຣມ. ການເລືອກ Go ແລະ Typescript ແມ່ນເພື່ອສຶກສາພາສາທີ່ທັນສະໄໝ, ມີຄວາມນິຍົມສູງ, ແລະ ມີການອອກແບບທີ່ແຕກຕ່າງກັນໃນດ້ານປະສິດທິພາບ ແລະ ການຈັດການ concurrency, ໂດຍສະເພາະໃນວຽກທີ່ກ່ຽວຂ້ອງກັບການຈັດການຖານຂໍ້ມູນຂອງ Web Application. ຜົນການຄົ້ນຄວ້າຈະຊ່ວຍໃຫ້ນັກພັດທະນາສາມາດຕັດສິນໃຈເລືອກພາສາທີ່ເໝາະສົມທີ່ສຸດກັບຄວາມຕ້ອງການຂອງການພັດທະນາລະບົບ Web Application ໃນອະນາຄົດ.

1.2. ຄຳຖາມຂອງການຄົ້ນຄ້ວາ

1) ລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ INSERT ຂໍ້ມູນ ເຂົ້າຖານຂໍ້ມູນ MySQL ໜ້ອຍກວ່າກັນ.
2) ລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ SELECT ຂໍ້ມູນ ຈາກຖານຂໍ້ມູນ MySQL ໜ້ອຍກວ່າກັນ.
3) ລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ UPDATE ຂໍ້ມູນ ເຂົ້າຖານຂໍ້ມູນ MySQL ໜ້ອຍກວ່າກັນ.
4) ລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ DELETE ຂໍ້ມູນ ຈາກຖານຂໍ້ມູນ MySQL ໜ້ອຍກວ່າກັນ.

1.3. ຈຸດປະສົງຂອງການຄົ້ນຄ້ວາ

1) ເພື່ອປຽບທຽບລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript ໃນການ INSERT ຂໍ້ມູນ ເຂົ້າຖານຂໍ້ມູນ MySQL.
2) ເພື່ອປຽບທຽບລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ SELECT ຂໍ້ມູນ ຈາກຖານຂໍ້ມູນ MySQL.
3) ເພື່ອປຽບທຽບລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ UPDATE ຂໍ້ມູນ ເຂົ້າຖານຂໍ້ມູນ MySQL.
4) ເພື່ອປຽບທຽບລະຫວ່າງພາສາໂປຣແກຣມ Go ແລະ Typescript, ພາສາໃດໃຊ້ເວລາໃນການ DELETE ຂໍ້ມູນ ຈາກຖານຂໍ້ມູນ MySQL.

2. ບົດຄົ້ນຄວ້າທີ່ກ່ຽວຂ້ອງ

2.1. ທິບທອນແນວຄິດ ແລະ ທິດສະດີທີ່ກ່ຽວຂ້ອງ

2.1.1. ທິດສະດີທີ່ກ່ຽວກັບ Web application

Web application ແມ່ນໂປຣແກຣມທີ່ເຮັດວຽກງານຜ່ານ Web Browser ເຊິ່ງສາມາດເຂົ້າເຖິງ ແລະ ເຮັດວຽກໄດ້ຈາກອຸປະກອນທີ່ເຊື່ອມຕໍ່ກັບອິນເຕີເນັດ ໂດຍບໍ່ຈຳເປັນຕ້ອງຕິດຕັ້ງໂປຣແກຣມເພີ່ມເຕີມລົງໃນຄອມພິວເຕີ ຫຼື ອຸປະກອນ ຂອງຜູ້ໃຊ້.

2.1.2. ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ ພາສາ Go

Golang ເປັນທີ່ຮູ້ຈັກຢ່າງເປັນທາງການໃນນາມ Go ເປັນພາສາຂຽນໂປຣແກຣມທີ່ຖືກອອກແບບໂດຍນັກວິສະວະກອນຂອງ Google Robert Griesemer, Rob Pike, ແລະ Ken Thompson ໄດ້ຮັບການອອກແບບມາໃຫ້ລຽບງ່າຍ, ມີປະສິດທິພາບ ແລະ ເຊື່ອຖືໄດ້ Go ເໝາະສຳລັບການພັດທະນາ application ສະໄໝໃໝ່ ໂດຍສະເພາະໃນດ້ານ server ແລະ ລະບົບໂຄງສ້າງພື້ນຖານສ່ວນຫຼັງ ໂດຍໄວຍະກອນທີ່ກົງໄປກົງມາ ການສະໜັບສະໜຸນໃນຕົວສຳລັບການເຮັດວຽກພ້ອມກັນ ແລະ ປະສິດທິພາບທີ່ດີເລີດ Go ໄດ້ກາຍເປັນຕົວເລືອກຍອດນິຍົມໃນໝູ່ນັກພັດທະນາສຳລັບການສ້າງ web application, micro service ແລະ ລະບົບແບບກະຈາຍ. Ecosystem ຂອງ Go ແມ່ນຂະຫຍາຍຕົວຢ່າງວ່ອງໄວ ນັບຕັ້ງແຕ່ເລີ່ມເປີດໂຕໃນ 2009 ໂດຍມີ libraries ແລະ ເຄື່ອງມືຫຼວງຫຼາຍ ໃຫ້ນັກພັດທະນາໄດ້ໃຊ້ປະໂຫຍດເປັນຕົ້ນແມ່ນ ບໍລິສັດຕ່າງໆເຊັ່ນ: Dropbox, Uber ແລະ Docker ໄດ້ເລືອກ Go ສຳລັບລະບົບ Backend ພື້ນຖານ ເຊິ່ງເນັ້ນໜັກເຖິງຄວາມສຳຄັນ ແລະ ຄວາມກ່ຽວຂ້ອງໃນສະພາບແວດລ້ອມທາງເທັກໂນໂລຢີໃນປະຈຸບັນ.

2.1.3. ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບພາສາ Typescript

Typescript ເປັນພາສາການຂຽນໂປຣແກຣມທີ່ພັດທະນາໂດຍ Microsoft ມັນມີ syntax ທີ່ເຄັ່ງຄັດສຳລັບການຂຽນປະເພດທີ່ຄວບຄຸມການນຳໃຊ້ປະເພດຂໍ້ມູນໃນໂຄງການ. Typescript ຖືກອອກແບບມາເພື່ອເຮັດໃຫ້ການຂຽນໂປລ

ແກລມໃນ JavaScript ຫຼາຍປະເພດຮູ້. ນີ້ແມ່ນເປັນປະໂຫຍດຫຼາຍສຳລັບການພັດທະນາລະບົບຂະໜາດໃຫຍ່ ແລະ Application ເນື່ອງຈາກວ່າມັນສາມາດຊ່ວຍກວດຫາຄວາມຜິດພາດກ່ອນທີ່ໂຄງການຈະຖືກປະຕິບັດ. ໃນ JavaScript, ພວກເຮົາຕ້ອງດຳເນີນການໂຄງການກ່ອນທີ່ຈະພົບຂໍ້ຜິດພາດໃນການນຳໃຊ້ປະເພດຂໍ້ມູນ Typescript ສາມາດຖືກນຳໃຊ້ເພື່ອພັດທະນາ Application JavaScript ທີ່ເຮັດວຽກທັງດ້ານ Client-side ແລະ Server-side ພວກເຮົາສາມາດໃຊ້ມັນເພື່ອພັດທະນາໂປຣແກຣມຕ່າງໆໃນ Node.js, Deno ຫຼື web browsers. ໃນທີ່ສຸດ, code ທີ່ຂຽນໃນ Typescript ຖືກແປເປັນ JavaScript ເພື່ອດຳເນີນການ ແລະ ນຳໃຊ້ເພື່ອປ່ຽນ code Typescript ເປັນ JavaScript, ພວກເຮົາຍັງສາມາດໃຊ້ເຄື່ອງມືເຊັ່ນ Babel ເພື່ອກຳນົດວິທີການລວບລວມໂປຣແກຣມ.

Typescript ຮອງຮັບໄຟລ໌ Declaration, ເຊິ່ງເປັນໄຟລ໌ທີ່ມີນາມສະກຸນ d.ts, ທີ່ຖືກນຳໃຊ້ເພື່ອກຳນົດປະເພດຕ່າງໆສຳລັບສ່ວນຕ່າງໆຂອງ code ເຊັ່ນ: ຕົວແປ, ຟັງຊັນ, ແລະ class ພວກມັນມັກຈະຖືກໃຊ້ເພື່ອອະທິບາຍປະເພດຂອງ code ທີ່ຂຽນໃນ JavaScript, ເພື່ອຊ່ວຍໃຫ້ code ເຮັດວຽກຢ່າງບໍ່ຢຸດຢັ້ງກັບ Typescript. ສຳລັບໂມດູນພື້ນຖານໃນ Node.js, ມີໄຟລ໌ Declaration ທີ່ມີຢູ່ສຳລັບການຂຽນ Typescript ໃນ Node.js.

2.1.4. ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ MySQL Database

SQL, ຫຍໍ້ມາຈາກ Structured Query Language, ແມ່ນພາສາການຂຽນໂປຣແກຣມທີ່ຊ່ວຍໃຫ້ເຮົາສາມາດສື່ສານກັບຖານຂໍ້ມູນ ທີ່ຮຽກຮ້ອງໃຫ້ມີການເຊື່ອມໂຍງເຄື່ອງມືອື່ນໆເພື່ອໃຫ້ໄດ້ລະບົບທີ່ຮອງຮັບຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ເຊັ່ນ: ເຮັດວຽກກັບ Web server ເພື່ອໃຫ້ບໍລິການສຳລັບ scripting languages ທີ່ເຮັດວຽກຢູ່ຝັ່ງເຄື່ອງບໍລິການເຊັ່ນ: PHP, aps.net, ແລະ ອື່ນໆ, ລວມທັງການລວບລວມຂໍ້ມູນຂອງຜູ້ເຂົ້າມາໃຊ້ງານຕ່າງໆ ປະຫວັດສາດ ຫຼື ຮູບພາບຕ່າງໆ ທີ່ພວກເຮົາເຫັນຢູ່ໃນເວັບໄຊທ໌ ຕ່າງກໍ່ຕ້ອງໃຊ້ພື້ນທີ່ເພື່ອເກັບຮັກສາ, ຊຶ່ງທັງໝົດທີ່ເກັບໄວ້ແລະ ສະແດງນັ້ນ ມີເບື້ອງລັກສະນະດຽວກັນກໍ່ຄື SQL.

2.1.5. ຄວາມໝາຍຂອງ ບຣາວເຊີ Browser

ເວັບບຣາວເຊີ ຫຼື ເອີ້ນຫຍໍ້ວ່າ ບຣາວເຊີ ແມ່ນຊ໊ອບແວແອັບພິຊັນ (software application) ໜຶ່ງໃຊ້ສຳລັບ ການເຂົ້າເຖິງ ແລະ ສະແດງເວັບໄຊທ໌ໃນອິນເຕີເນັດ ໂດຍບຣາວເຊີ ຈະສົ່ງຄຳຮ້ອງຂໍໄປຫາ server ທີ່ໃຫ້ບໍລິການເວັບໄຊທ໌ ແລະ ສະແດງຂໍ້ມູນທີ່ໄດ້ຮັບ ໃນຮູບແບບທີ່ເຂົ້າໃຈໄດ້ ໃຫ້ກັບຜູ້ໃຊ້ ຜ່ານ hyTypescript ink ແລະ ເຮັດໜ້າທີ່ເຊື່ອມໂຍງບັນດາໜ້າເວັບທີ່ນອນຢູ່ໃນເວັບໄຊທ໌ອັນດຽວກັນ ຫຼື ທີ່ຢູ່ເວັບໄຊທ໌ ທີ່ຕ່າງກັນ ການເອີ້ນຂໍ້ມູນຂອງບຣາວເຊີ ແມ່ນຈະເອີ້ນໃນຮູບແບບ HTML ເພື່ອສະແດງຜົນໃນໜ້າເວັບໄຊທ໌ໜຶ່ງອາດສະແດງຜົນແຕກຕ່າງກັນໃນໜ້າ ບຣາວເຊີ ທີ່ຕ່າງກັນ; ເວັບໄຊທ໌ທີ່ນິຍົມໃຊ້ກັນຫຼາຍມີຄື: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari ແລະ Internet Explorer.

2.2. ທົບທວນເອກະສານການຄົ້ນຄວ້າທີ່ກ່ຽວຂ້ອງ

ທ່ານ Priyanka Gowda Ashwath Narayana Gowda, ໄດ້ເຮັດການວິໄຈກ່ຽວກັບ ການວິເຄາະປຽບທຽບ ພາສາໂປຣແກຣມ Typescript ປຽບທຽບກັບ ພາສາໂປຣແກຣມ JavaScript ຂອງການຄົ້ນຄວ້ານີ້ແມ່ນໄວ້ວ່າທັງ JavaScript ແລະ Typescript ແມ່ນພາສາທີ່ດີເລີດ ທີ່ມີຄວາມສາມາດທີ່ແຕກຕ່າງກັນແຕ່ຢ່າງໃດກໍຕາມ ເຖິງແມ່ນວ່າ javascript ຍັງຄົງເປັນພາສາທີ່ໃຊ້ງານໄດ້ຫຼາກຫຼາຍ ແຕ່ Typescript ກໍ່ໄດ້ບັນລຸຈຸດປະສົງໃນການແກ້ໄຂຂໍ້ບົກຜ່ອງຂອງ javascript. ສາມາດສະຫຼຸບວ່າ Typescript ແມ່ນສະໜອງຜົນປະໂຫຍດຫຼາຍກວ່າເມື່ອເວົ້າເຖິງໂຄງການຂະໜາດໃຫຍ່ ແລະ ສະລັບສັບຊ້ອນ, ນີ້ແມ່ນຍ້ອນວ່າໃຊ້ວິທີການການພິມແບບຄົງທີ່ ທີ່ເຮັດວຽກໄດ້ດີກັບການເຊື່ອມໂຍງ IDE, ເຮັດໃຫ້ສະດວກສະບາຍສຳລັບໂຄງການຮ່ວມມື ເປັນສິ່ງຈຳເປັນທີ່ນັກພັດທະນາ ແລະ ອົງການຈະຕ້ອງມີຄວາມຄຸ້ນເຄີຍກັບຂໍ້ດີ ແລະ ຂໍ້ເສຍຂອງແຕ່ລະພາສາ, ປະເມີນຄວາມຕ້ອງການຂອງຕົນ ແລະ ເລືອກທາງເລືອກທີ່ເຮັດວຽກທີ່ດີທີ່ສຸດສຳລັບຕົນເອງບໍ່ຈຳເປັນຈຳກັດພຽງທາງເລືອກດຽວ. ເພື່ອໃຊ້ຄຸນສົມບັດທີ່ດີທີ່ສຸດຂອງທັງສອງພາສາເພື່ອເພີ່ມປະສິດທິພາບການເຮັດວຽກ ແລະ ເພີ່ມຄຸນນະພາບທາງດ້ານປະສິດທິພາບສູງສຸດຂອງຜົນທີ່ຕ້ອງການໄດ້ຮັບ (Priyanka, 2019).

ທ່ານ Lee ແລະ Lim, ໄດ້ເຮັດການວິໄຈກ່ຽວກັບ ການປະມວນຜົນການປຽບທຽບບັນຊີລາຍການຂໍ້ມູນຂະໜາດໃຫຍ່ ໂດຍໃຊ້ພາສາ GO ຈາກຜົນການວິໄຈສະແດງໃຫ້ເຫັນວ່າ ການປະມວນຜົນຂໍ້ມູນຈຳນວນຫຼາຍໂດຍໃຊ້ພາສາ GO ແລະ ປ່ຽນເປັນໂຄງສ້າງ JSON ໃນຮູບແບບຂອງລາຍຊື່ຂໍ້ມູນ ການປະມວນຜົນຂໍ້ມູນຈຳນວນຫຼວງຫຼາຍໃນຮູບແບບທີ່ໃຊ້ໃນ Python ຂັ້ນພື້ນຖານຜົນການປະມວນຜົນຈະສົ່ງອອກເປັນລາຍການໃນຮູບແບບ JSON ແລະ ໄດ້ກວດເບິ່ງວ່າມັນແຕກຕ່າງກັນແນວໃດ. ໃນ Python ຄວາມໄວຂອງການຄົ້ນຫາໂດຍເອີ້ນ PostgreSQL ແມ່ນໄວກວ່າພາສາ GO ເຮົາສາມາດເຫັນຜົນໄດ້ຮັບໄວ ແຕ່ໃນຮູບແບບ JSON ແປງ ແລະ ປ່ຽນປະເພດຂໍ້ມູນຕາຕະລາງໃນ PostgreSQL ຜົນໄດ້ຮັບຂອງການນຳໃຊ້ ແລະ ການປະມວນຜົນຟັງຊັນເພື່ອຈຸດປະສົງນີ້ ຄ່າຈະຖືກປະມວນຜົນໄວຂຶ້ນເລັກນ້ອຍ ໂດຍໃຊ້ການເຮັດວຽກພ້ອມກັນຂອງພາສາ GO. ມັນໄດ້ຮັບການປະມວນຜົນໄວຂຶ້ນ. ຜົນການທົດລອງຊີ້ໃຫ້ເຫັນວ່າບາງຟັງຊັນພິເສດ ຄວາມໄວການປະມວນຜົນ ອາດຈະແຕກຕ່າງກັນ ຂຶ້ນຢູ່ກັບພາສາການພັດທະນາ ຟັງຊັນທີ່ປະມວນຜົນໂດຍໃຊ້ພາສາ GO ໄວກວ່າພາສາສະເພາະອື່ນໆ ທີ່ຖືກນຳໃຊ້ແບບຕໍ່ເນື່ອງເລັກນ້ອຍ ອາດເຮັດໃຫ້ເກີດຜົນໄດ້ຮັບການປະມວນຜົນທີ່ແຕກຕ່າງກັນບາງຢ່າງອື່ນໆ ເຫັນໄດ້ວ່າມີວິທີການປະມວນຜົນທີ່ຄ້າຍຄືກັນກັບພາສາ GO ໃນພາສາການພັດທະນາຢ່າງຊັດເຈນ. ເຖິງຢ່າງໃດກໍ່ຕາມ, ເນື່ອງຈາກຄຸນລັກສະນະຂອງພາສາ GO, ຟັງຊັນທີ່ສ້າງຂຶ້ນຖືກ

ເອີ້ນວ່າ GO ການປະມວນຜົນຊ່ວຍໃຫ້ເຮົາສາມາດປະມວນຜົນ ຂະບວນການພ້ອມໆກັນໄດ້ຢ່າງງ່າຍດາຍ. ຖ້າ ຫາກ ວ່າ ນຳ ໃຊ້ ຄຸນ ນະ ສົມ ບັດການເຮັດວຽກພ້ອມກັນ ເຮົາ ສາ ມາດ ປະມວນ ຜົນ ທັງ ໝົດ ໃນ ເວ ລາ ດຽວ ກັນ ໄດ້ ເປັນວິທີການສຳລັບການ ປະມວນຜົນປະລິມານຂະໜາດໃຫຍ່ຂອງຖານຂໍ້ມູນນັກຄົ້ນຄວ້າ ຄິດວ່າມັນຈະເປັນການພັດທະນາທີ່ດີ (Lee & Lim, 2022).

ທ່ານ Marci ແລະ Małgorzata, ໄດ້ເຮັດບົດວິໄຈ ກ່ຽວກັບການວິເຄາະປຽບທຽບຂອງກອບການເຮັດວຽກໂດຍໃຊ້ Typescript ເພື່ອສ້າງແອັບພລິເຄຊັນເຊີຟເວີເຫັນວ່າ ຈຸດປະສົງ ຂອງບົດຄວາມນີ້ແມ່ນເພື່ອປຽບທຽບການປະຕິບັດຂອງກອບການ ຂຽນໂປຣແກຣມທີ່ໃຊ້ໃນການຂຽນແອັບພລິເຄຊັນເຊີຟເວີ ໂດຍ ໃຊ້ພາສາການຂຽນໂປຣແກຣມ Typescript. ປັດໄຈທີ່ສຳຄັນທີ່ ສຸດທີ່ພິຈາລະນາໃນການຄົ້ນຄວ້າແມ່ນເວລາຕອບສະໜອງຕໍ່ການ ຮ້ອງຂໍ, ເພາະວ່າມັນເປັນພື້ນຖານສຳລັບການກຳນົດປະສິດທິພາບ ຂອງແອັບພລິເຄຊັນ. ການຈຳລອງສະພາບການເຮັດວຽກຕາມທຳ ມະຊາດເຊິ່ງມີການໃຊ້ເຕັກໂນໂລຢີທີ່ປຽບທຽບນັ້ນລວມຢູ່ໃນ ສະຖານະການການຄົ້ນຄວ້າທີ່ຖືກສ້າງຂຶ້ນ. ການທົດລອງ ANOVA ແລະ Tukey ສະແດງໃຫ້ເຫັນຄວາມແຕກຕ່າງທີ່ມີ ຄວາມສຳຄັນທາງສະຖິຕິໃນໄລຍະເວລາປຽບທຽບທີ່ໄດ້ຮັບໂດຍ skeletons. ອີງຕາມຜົນໄດ້ຮັບທີ່ໄດ້ຮັບ, ມັນສາມາດບອກໄດ້ຢ່າງ ຊັດເຈນວ່າຂະໜາດຂອງຂໍ້ມູນທີ່ຖືກໂອນມີຜົນກະທົບຕໍ່ເວລາ ຕອບສະໜອງ. ນີ້ແມ່ນເຫັນໄດ້ຊັດເຈນທີ່ສຸດໃນເວລາທີ່ຈັດການ ກັບຄຳຮ້ອງຂໍ GET. ໃນຂະນະດຽວກັນ, ຍ້ອນວ່າຈຳນວນວັດຖຸທີ່ ສົ່ງຄືນເພີ່ມຂຶ້ນ, ຄວາມແຕກຕ່າງລະຫວ່າງໂຄງກະດູກທີ່ສົມທຽບ ເພີ່ມຂຶ້ນ. ໃນການວັດແທກໂດຍອີງໃສ່ຄຳຮ້ອງຂໍ GET, NestJS ປະຕິບັດໄດ້ດີທີ່ສຸດ, ບັນລຸເວລາການບໍລິການທີ່ສັ້ນທີ່ສຸດ. ຊ້າທີ່ ສຸດສຳລັບທຸກຂະໜາດຂໍ້ມູນແມ່ນ Ts.ED ແລະ ຄວາມແຕກຕ່າງ ກັບຄູ່ແຂ່ງອື່ນໆໃນການປຽບທຽບຕໍ່ມາໄດ້ເພີ່ມຂຶ້ນເຖິງຂໍ້ເສຍຂອງ ກອບການເຮັດວຽກນີ້. ທີ່ໜ້າສົນໃຈຄື FoalTS ຊ່ວຍຫຼຸດຜ່ອນ ຄວາມແຕກຕ່າງເມື່ອທຽບກັບ NestJS ທີ່ມີຂະໜາດຂໍ້ມູນໃຫຍ່ ກວ່າ, ແລະ ເມື່ອສົ່ງຄືນວັດຖຸ 5000 ແລະ 10000, ບໍ່ມີຄວາມ ແຕກຕ່າງທາງສະຖິຕິ. ສຳລັບການຮ້ອງຂໍ POST, PUT, ແລະ DELETE, NestJS ເຂົ້າມາເປັນອັນດັບທຳອິດ, ໂດຍ FoalTS ແລະ Ts.ED ເຂົ້າມາໃນອັນທີສອງ ແລະ ທີສາມ, ຕາມລຳດັບ. ຫຼັງ ຈາກດຳເນີນການວິເຄາະ, ມັນສາມາດສະຫຼຸບໄດ້ວ່າກອບທີ່ມີ ປະສິດທິພາບທີ່ສຸດໃນບັນດາການປຽບທຽບທັງໝົດແມ່ນ NestJS, ໃນຂະນະທີ່ FoalTS ແລະ Ts.ED ໄດ້ອັນດັບສອງ ແລະ ທີສາມຕາມລຳດັບ (Marci&Małgorzata, 2022).

ທ່ານ Smith (2022) ໄດ້ເຮັດບົດວິໄຈກ່ຽວກັບການ ປຽບທຽບປະສິດທິພາບການວິເຄາະຂອງ Go ແລະ Node.js ສຳ ລັບ Web Services ທີ່ມີການເຮັດວຽກພ້ອມກັນສູງ ຈາກຜົນ ການວິໄຈສະແດງໃຫ້ເຫັນວ່າ ການສຶກສາປຽບທຽບປະສິດທິພາບ ຂອງ Go ແລະ Node.js (ເຊິ່ງ Typecript ຖືກ Compile ລົງ) ໃນການຈັດການ Web Services ທີ່ມີ Concurrency ສູງ. ຜົນ ການວິໄຈສະແດງໃຫ້ເຫັນວ່າ Go ມັກຈະມີການໃຊ້ ຊັບພະຍາກອນ (CPU, Memory) ທີ່ຕ່ຳກວ່າ ແລະ ໃຫ້ Throughput (ຈຳນວນຄຳຮ້ອງຂໍທີ່ຈັດການໄດ້ຕໍ່ວິນາທີ) ທີ່ສູງ ກວ່າ ໃນສະຖານະການທີ່ມີການໂຫຼດສູງ, ໂດຍສະເພາະເມື່ອມີ ການປະມວນຜົນທີ່ຮຽກຮ້ອງ CPU ຫຼາຍ. ຢ່າງໃດກໍຕາມ, Node.js ຍັງຄົງມີຄວາມໄດ້ປຽບໃນດ້ານການພັດທະນາທີ່ ວ່ອງໄວສຳລັບ I/O-bound applications.

ທ່ານ Chen & Wang (2021) ໄດ້ເຮັດບົດວິໄຈກ່ຽວ ກັບການປະເມີນປະສິດທິພາບການເຂົ້າເຖິງຖານຂໍ້ມູນ ປະສິດທິພາບການເຂົ້າເຖິງຖານຂໍ້ມູນ ຂອງ Web Frameworks ທີ່ນິຍົມ: ກໍລະນີສຶກສາ Go ແລະ Express.js. ຜົນການວິໄຈຊີ້ ໃຫ້ເຫັນວ່າໃນການດຳເນີນງານຖານຂໍ້ມູນພື້ນຖານ (CRUD), Go ມັກຈະມີ Latency (ເວລາຕອບສະໜອງ) ທີ່ຕ່ຳກວ່າ ເນື່ອງ ຈາກການຈັດການ Concurrency ທີ່ດີກວ່າ ແລະ ໂຄງສ້າງພາສາ ທີ່ເໝາະສົມກັບ System-level programming ແຕ່ສຳລັບວຽກ ທີ່ຊັບຊ້ອນກວ່າ, ຄວາມແຕກຕ່າງອາດຈະຂຶ້ນຢູ່ກັບການອອກ ແບບຖານຂໍ້ມູນ ແລະ ORM/Driver ທີ່ໃຊ້.

ເຖິງແມ່ນວ່າບົດຄົ້ນຄວ້າທີ່ກ່າວມາຂ້າງເທິງນັ້ນຈະໃຫ້ ຄວາມເຂົ້າໃຈທີ່ສຳຄັນກ່ຽວກັບ Go ແລະTypeScript JavaScript ໃນດ້ານຕ່າງໆກໍ່ຕາມ ແຕ່ບົດຄົ້ນຄວ້າຂອງພວກເຮົາ ກໍ່ມີຄວາມແຕກຕ່າງຢ່າງຊັດເຈນເຊັ່ນ: ການປຽບທຽບລະຫວ່າງ Go ແລະ Typescript ແມ່ນຈະເນັ້ນສະເພາະໃນ Web Application Backend. ໃນບົດວິໄຈຂອງ ທ່ານ Priyanka Gowda ແມ່ນເນັ້ນໃສ່ JavaScript vs. TypeScript ເປັນຫຼັກ, ໂດຍບໍ່ໄດ້ລວມ Go ເຂົ້າໃນການປຽບທຽບປະສິດທິພາບ. ສຳລັບ ບົດວິໄຈການປຽບທຽບພາສາ Go ກັບ ພາສາ Python ຂອງ ຂອງ ທ່ານ Lee & Lim ແມ່ນສຸມໃສ່ການປະມວນຜົນຂໍ້ມູນຈຳນວນ ຫຼາຍເປັນ JSON, ແຕ່ບໍ່ໄດ້ປຽບທຽບ Go ກັບ TypeScript ໃນ Web Application backend ແລະ ບົດວິໄອການປຽບທຽບ Frameworks ຂອງ TypeScript ເພື່ອສ້າງ Server Applications ຂອງທ່ານ Marci & Małgorzata ແຕ່ບໍ່ໄດ້ ປຽບທຽບ TypeScript ກັບ Go. ບົດຄົ້ນຄວ້າຂອງທ່ານ Smith (2022) ປຽບທຽບ Go ກັບ Node.js (ເຊິ່ງ TypeScript Compile ລົງ), ແຕ່ຈະເນັ້ນການປຽບທຽບໃນລະດັບສູງ (High-Concurrency Web Services), ບໍ່ໄດ້ເຈາະຈົງເຖິງ ການ ຈັດການກັບຖານຂໍ້ມູນ (CRUD) ຢ່າງລະອຽດ ມີພຽງບົດ ຄົ້ນຄວ້າຂອງທ່ານ Chen & Wang (2021) ທີ່ໃກ້ຄຽງກວ່າໂດຍ ຈະສຸມໃສ່ປະສິດທິພາບການເຂົ້າເຖິງຖານຂໍ້ມູນ, ແຕ່ກໍຍັງເນັ້ນໃສ່ Frameworks ເປັນຫຼັກ, ບໍ່ແມ່ນກ່ຽວກັບພາສາໂປຣແກຣມໂດຍ

ກົງ. ສ່ວນປິດຄົ້ນຄວ້າຂອງເຮົາແມ່ນຈະເຈາະຈົງໃສ່ເວລາໃນການຕອບສະໜອງ (Response Time) ແລະ ປະສິດທິພາບໃນການປະມວນຜົນຂໍ້ມູນ ຂອງ Go ແລະ TypeScript ເມື່ອປະຕິບັດງານ CRUD (Select, Insert, Update, Delete) ກັບຖານຂໍ້ມູນ, ເຮົາຈະທົດລອງກັບ ຈຳນວນຂໍ້ມູນທີ່ແຕກຕ່າງກັນ ເລີ່ມແຕ່ 10 ຫາ 100,000 ແຖວ, ເຊິ່ງຈະຊ່ວຍໃຫ້ເຫັນພາບທີ່ຊັດເຈນກ່ຽວກັບປະສິດທິພາບຂອງແຕ່ລະພາສາພາຍໃຕ້ສະຖານະການໂຫຼດຂໍ້ມູນທີ່ເພີ່ມຂຶ້ນ, ຕັ້ງແຕ່ຂໍ້ມູນຂະໜາດນ້ອຍຈົນເຖິງຂະໜາດກາງ ແລະ ດຳເນີນການໃນສະພາບແວດລ້ອມທີ່ຄວບຄຸມໄດ້ໂດຍຕິດຕັ້ງ Web Application ໄວ້ຝັ່ງ Server ແລະ Client ເພື່ອດຳເນີນການທົດລອງ, ເຊິ່ງຈະຊ່ວຍຫຼຸດຜ່ອນຕົວແປພາຍນອກ ແລະ ຮັບປະກັນຄວາມໜ້າເຊື່ອຖືຂອງຜົນການທົດລອງ.

3. ວິທີດຳເນີນການຄົ້ນຄວ້າ

3.1 ປະຊາກອນ ແລະ ກຸ່ມຕົວຢ່າງ

ໃນການສຶກສາຄົ້ນຄວ້ານີ້, ພວກເຮົາໄດ້ກຳນົດປະຊາກອນ ແລະ ກຸ່ມຕົວຢ່າງດັ່ງລຸ່ມນີ້:

3.1.1 ປະຊາກອນ (Population)

ປະຊາກອນໃນການຄົ້ນຄວ້ານີ້ ແມ່ນ Web Application ທີ່ຖືກພັດທະນາໂດຍນຳໃຊ້ພາສາໂປຣແກຣມ Go (Golang) ແລະ Typescript ໃນສະພາບແວດລ້ອມການທົດລອງດຽວກັນ, ໂດຍມີການເຊື່ອມຕໍ່ກັບຖານຂໍ້ມູນແບບດຽວກັນ.

3.1.2 ກຸ່ມຕົວຢ່າງ (Sample)

ກຸ່ມຕົວຢ່າງທີ່ຖືກນຳໃຊ້ໃນການຄົ້ນຄວ້ານີ້ແມ່ນ: Web Application ທີ່ຖືກພັດທະນາດ້ວຍ ພາສາ Go (Golang) ແລະ Web Application ທີ່ຖືກພັດທະນາດ້ວຍ ພາສາ Typescript.

ທັງສອງ Web Application ນີ້ ໄດ້ຖືກສ້າງຂຶ້ນມາເພື່ອປະຕິບັດໜ້າທີ່ຫຼັກດຽວກັນຄື: ການຈັດການຖານຂໍ້ມູນ ໃນການ INSERT, SELECT, UPDATE, ແລະ DELETE. ພວກມັນຖືກນຳໄປທົດລອງໃນສະພາບແວດລ້ອມທີ່ຄືກັນ ແລະ ພາຍໃຕ້ເງື່ອນໄຂການໂຫຼດຂໍ້ມູນທີ່ແຕກຕ່າງກັນ (10, 100, 1,000, 10,000, ແລະ 100,000 ແຖວ) ເພື່ອວັດແທກຄວາມໄວໃນການຕອບສະໜອງ.

3.2 ເຄື່ອງມືທີ່ໃຊ້ໃນການເກັບກຳຂໍ້ມູນ

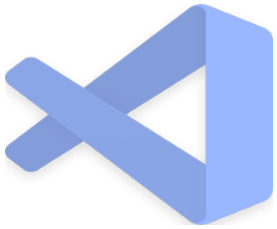
3.2.1 ອຸປະກອນຮາດແວ (Hardware)

ການທົດລອງດຳເນີນການຢູ່ໃນສະພາບແວດລ້ອມທີ່ຄວບຄຸມໄດ້ ໂດຍໃຊ້ອຸປະກອນຄອມພິວເຕີ້ແລັບທັອບທີ່ມີຂໍ້ກຳໜົດດັ່ງນີ້:

- ເຄື່ອງຄອມພິວເຕີ້: Acer Laptop
- ລະບົບປະຕິບັດການ: Windows 10 Professional
- ຕົວປະມວນຜົນ: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz (Turbo Boost ເຖິງ 2.70 GHz)
- ໜ່ວຍຄວາມຈຳ: 16 GB RAM
- ອຸປະກອນເກັບຂໍ້ມູນ: SSD 120 GB

3.2.2 ອຸປະກອນ Software

- Visual Studio Code: Version 1.93.1 ເປັນ Integrated Development Environment (IDE) ໃນການຂຽນ ແລະ ທົດລອງໂຄດ
- Go Programming Language: Version go1.22.5 windows/amd64
- Typescript Programming Language: Version 5.8.3
- Node.js: Version v22.15.0 - ສຳລັບການດຳເນີນງານ Typescript runtime
- Node Package Manager (NPM): Version 11.4.0 - ສຳລັບການຈັດການ dependencies
- ລະບົບຈັດການຖານຂໍ້ມູນ MySQL: Version 10.4.17-MariaDB (mariadb.org binary distribution)
- XAMPP: ສຳລັບການສ້າງສະພາບແວດລ້ອມເຊີເວີ້ທ້ອງຖິ່ນ



ຮູບທີ 1: ໂປຣແກຣມ Visual Studio Code. ແຫຼ່ງຂໍ້ມູນຈາກ https://webcatalog.io/en/apps/vs-code

3.3 ວິທີເກັບກຳຂໍ້ມູນ

ສຳລັບວິທີການທີ່ຈະນຳມາໃຊ້ ໃນການເກັບຂໍ້ມູນແມ່ນມາຈາກຜົນການທົດລອງຕົວຕົວຈິງ ເຊິ່ງເລີ່ມຈາກການປ້ອນຂໍ້ມູ ນ (First Name, Last Name, Birth Date, E-mail, Phone) ຢູ່ຟອມໃນ Web application ແລະ ປ້ອນຄ່າຈຳນວນແຖວທີ່ຕ້ອງການ Select, Insert, Update, Delete ໂດຍສາມາດເອົາຂໍ້ມູນຈາກ ວິທີການວັດແທກ ແລ້ວເກັບໄວ້ໃນຟາຍ Excel. ອີງໃສ່ຕາຕະລາງທີ (1),(2) ຄວາມໄວໃນການປະມວນຜົນຂອງຂໍ້ມູນ ໂດຍຄິດໄລ່ເປັນ (ms).

❖ **Code ປ້ອນ ຈຳນວນແຖວທີ່ຕ້ອງການ:**

```
<div class="space-y-2">
    <label for="bulkCount" class="block text-sm
font-semibold text-gray-700">
        <i class="fas fa-hashtag mr-2 text-purple-
500"></i>ຈຳນວນແຖວທີ່ຕ້ອງການ *
            </label>
             <input type="number"
                    id="bulkCount"
                    min="1"
```

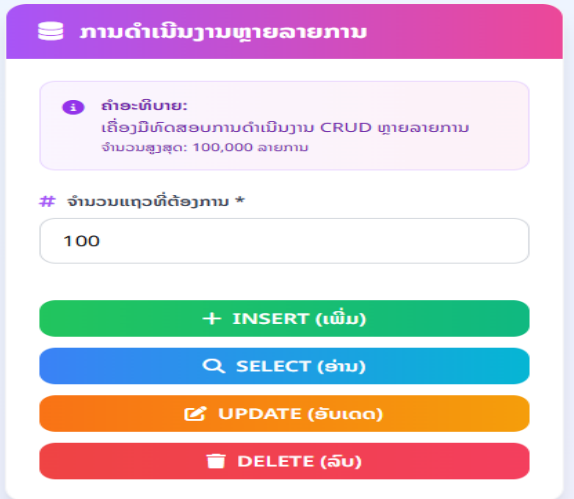
```
                    max="100000"
                    value="100"
                    class="w-full px-4 py-3
border border-gray-300 rounded-xl focus:ring-2
focus:ring-purple-500 focus:border-transparent
transition-all duration-200 placeholder-gray-400"
placeholder="ເຊັ່ນ: 100">
</div>
```

- **Code ປຸ່ມ INSERT, SELECT, UPDATE, DELETE:**

```
<div class="space-y-3 pt-4">
                <button type="button"
                    id="bulkInsertBtn"
                    class="w-full bg-gradient-
to-r from-green-500 to-emerald-500 hover:from-
green-600 hover:to-emerald-600 text-white font-
semibold py-2 px-4 rounded-xl transition-all
duration-200 transform hover:scale-105
hover:shadow-lg">
                    <i class="fas fa-plus mr-
2"></i>INSERT (ເພີ່ມ)
                </button>
                <button type="button"
                    id="bulkSelectBtn"
                    class="w-full bg-gradient-
to-r from-blue-500 to-cyan-500 hover:from-blue-600
hover:to-cyan-600 text-white font-semibold py-2 px-
4 rounded-xl transition-all duration-200 transform
hover:scale-105 hover:shadow-lg">
                    <i class="fas fa-search mr-
2"></i>SELECT (ອ່ານ)
                </button>
                <button type="button"
                    id="bulkUpdateBtn"
                    class="w-full bg-gradient-
to-r from-orange-500 to-amber-500 hover:from-
orange-600 hover:to-amber-600 text-white font-
semibold py-2 px-4 rounded-xl transition-all
duration-200 transform hover:scale-105
hover:shadow-lg">
                    <i class="fas fa-edit mr-
2"></i>UPDATE (ອັບເດດ)
                </button>
                <button type="button"
                    id="bulkDeleteBtn"
                    class="w-full bg-gradient-
to-r from-red-500 to-rose-500 hover:from-red-600
hover:to-rose-600 text-white font-semibold py-2 px-4
rounded-xl transition-all duration-200 transform
hover:scale-105 hover:shadow-lg">
    <i class="fas fa-trash mr-2">
</i>DELETE (ລຶບ)
</button>
</div>
```

- **Code ຜົນການດຳເນີນງານ:**

```
<!-- Results Display -->
                <div id="bulkResults"
class="hidden">
                    <div class="bg-gradient-to-r
from-green-50 to-emerald-50 border border-green-
200 rounded-xl p-4">
                        <div class="flex items-start">
                            <i class="fas fa-chart-bar
text-green-600 mr-3 mt-1"></i>
                            <div class="text-green-800
text-sm">
                                <p class="font-semibold
mb-2">ຜົນການດຳເນີນງານ:</p>
                                <div
id="bulkResultContent" class="space-y-1">
                                    <!-- Results will be
inserted here -->
                                </div>
                            </div>
                        </div>
                    </div>
```



ຮູບທີ 2: ການສະແດງປຸ່ມປ້ອນຈຳນວນແຖວ ແລະ ປຸ່ມ INSERT, SELECT, UPDATE, DELETE. ສ້າງໂດຍຜູ້ຄົ້ນຄວ້າ

ตาຕະລາງ 1 : ເກັບກໍ່ຂໍ້ມູນຜົນການທົດລອງພາສາໂປຣແກຣມ
Go

จำนวนแถว	Operation	Go					
		#1	#2	#3	#4	#5	ສະເລ່ຍ
10	SELECT	112.23	524.61	507.59	510.74	565.22	444.08
	INSERT	8.23	8.02	10.93	10.01	10.37	9.51
	UPDATE	1.42	1.13	1.23	1.20	1.00	1.20
	DELETE	392.32	401.59	434.62	443.03	423.19	418.95
100	SELECT	563.32	597.41	633.56	629.28	611.01	606.92
	INSERT	55.46	31.60	40.97	21.15	38.78	37.59
	UPDATE	1.34	996.29	985.10	866.61	1.03	570.07
	DELETE	357.87	381.64	385.84	359.18	360.99	369.10
1,000	SELECT	501.56	504.40	542.38	541.38	494.50	516.84
	INSERT	192.63	80.31	75.25	76.35	65.45	98.00
	UPDATE	1.23	1.37	987.18	1.05	996.63	397.49
	DELETE	367.46	477.57	465.58	450.49	388.83	429.99
10,000	SELECT	558.54	540.06	533.41	543.56	573.83	549.88
	INSERT	817.50	810.45	765.30	770.25	655.15	763.73
	UPDATE	22.31	30.09	26.41	26.12	24.68	25.92
	DELETE	749.13	820.45	589.13	498.10	703.17	672.00
100,000	SELECT	688.09	717.98	746.56	764.03	781.91	739.71
	INSERT	5142.00	4950.00	4550.00	3900.00	3650.00	4438.40
	UPDATE	191.67	193.72	194.64	187.80	192.52	192.07
	DELETE	8.33	1.93	4.25	0.51	1.51	3.31

ตาຕະລາງ 2 : ເກັບກໍ່ຂໍ້ມູນຜົນການທົດລອງພາສາໂປຣແກຣມ
Typescript

จำนวนแถว	Operation	TypeScript					
		#1	#2	#3	#4	#5	ສະເລ່ຍ
10	SELECT	156.00	82.00	44.00	43.00	39.00	72.80
	INSERT	33.00	14.00	21.00	14.00	14.00	19.20
	UPDATE	114.00	104.00	97.00	122.00	101.00	107.60
	DELETE	13.41	11.26	1.03	937.00	963.00	385.14
100	SELECT	142.00	44.00	46.00	46.00	48.00	65.20
	INSERT	66.00	82.00	78.00	71.00	53.00	70.00
	UPDATE	90.00	99.00	100.00	103.00	97.00	97.80
	DELETE	1.06	978.00	906.00	1.04	973.00	571.82
1,000	SELECT	53.00	54.00	85.00	82.00	56.00	66.00
	INSERT	298.00	240.00	202.00	260.00	330.00	266.00
	UPDATE	209.00	190.00	164.00	176.00	186.00	185.00
	DELETE	982.00	1.07	1.08	1.02	1.04	197.24
10,000	SELECT	122.00	151.00	118.00	109.00	139.00	127.80
	INSERT	853.00	656.00	864.00	1.13	1.03	475.03
	UPDATE	446.00	397.00	340.00	313.00	310.00	361.20
	DELETE	1.31	1.19	1.14	1.06	1.30	1.20
100,000	SELECT	808.00	908.00	946.00	905.00	984.00	910.20
	INSERT	4284.00	7848.00	8311.00	8417.00	8342.00	7440.40
	UPDATE	617.00	219.35	235.65	242.35	256.32	314.13
	DELETE	13.38	21.17	35.44	24.43	7.87	20.45

3.4 ການວິເຄາະຂໍ້ມູນ

ການວິເຄາະ ແລະ ການຕີຄວາມໝາຍຂໍ້ມູນແມ່ນພວກເຮົາຈະສົມທຽບຈາກການນຳໃຊ້ຂໍ້ມູນຕົວຈິງທີ່ໄດ້ຮັບຈາກການທົດລອງໃນ 2 ຮູບແບບມາໃຊ້ເຂົ້າໃນການທົດລອງລະບົບ, ແລ້ວຈະນຳມາວິເຄາະຫາຄ່າສະເລ່ຍຂອງຂໍ້ມູນທີ່ໄດ້ເຮັດການທົດລອງນັ້ນຄື: ຄ່າຄວາມໄວໃນການປະມວນຜົນຂອງຂໍ້ມູນ, ການກຳນົດຈຳນວນຄັ້ງຂອງການທົດລອງແມ່ນ ຈະເຮັດການທົດລອງໂດຍການຂຽນ code ຄຳສັ່ງເອີ້ນໃຊ້ລະຫ່ວາງ ພາສາໂປຣແກຣມ Go ແລະ ພາສາໂປຣແກຣມ Typescript; ຈາກເຄື່ອງຜູ້ໃຊ້ໄປຫາ Server ແມ່ນຈະມີ 2 ຊ່ວງ, ຊ່ວງລະ 5 ຄັ້ງ ລວມທັງໝົດເປັນ 10 ຄັ້ງ. ຈາກຜົນການທົດລອງໃນການສົ່ງ ການເອີ້ນໃຊ້ ຈາກເຄື່ອງຜູ້ໃຊ້ ໂດຍໄດ້ສັງເກດການທົດລອງດ້ວຍການບັນທຶກຂໍ້ມູນທີ່ສາມາດບັນທຶກໄດ້ເຊັ່ນ: ຄ່າຄວາມໄວໃນການປະມວນຜົນຂອງຂໍ້ມູນ ເພີ່ມຂໍ້ມູນເຂົ້າໃນຖານຂໍ້ມູນ MySQL ໂດຍສະເລ່ຍທັງໝົດ 2 ຊ່ວງ. ສະນັ້ນຜູ້ຄົ້ນຄວ້າຈຶ່ງກຳນົດຈຳນວນຄັ້ງຂອງການທົດລອງໃນແຕ່ລະກໍລະນີ ຈຳນວນ 5 ຄັ້ງເທົ່ານັ້ນ.

❖ **ການຄຳນວນຫາຄ່າສະເລ່ຍ:**

$$\bar{x} = \frac{\sum x}{n} \qquad (3.1)$$

- $\bar{x}$ ແມ່ນຄ່າສະເລ່ຍ.
- $\sum x$ ແມ່ນຜົນລວມຂອງຂໍ້ມູນທັງໝົດ.ແມ່ນຈຳນວນຂອງຂໍ້ມູນ.

❖ **ການຄຳນວນຫາຄ່າແຕກຕ່າງ (%):**

$$\text{ເປີເຊັນ (\%)} = \left(\frac{\text{(ຄ່າທີ່ຕ້ອງການຫາ)}}{\text{ຄ່າທັງໝົດ}}\right) \times \mathbf{100} \quad \Rightarrow \quad \boldsymbol{x = \frac{A-B}{A \text{ or } b} \times 100} \qquad (3.2)$$

- x ແມ່ນ ເປີເຊັນ (%) ຂອງຄວາມແຕກຕ່າງ.
- A ແມ່ນຄ່າທີ 1.
- B ແມ່ນຄ່າທີ 2.

4. ຜົນການຄົ້ນຄວ້າ

4. 1 ຄວາມໄວໃນການປະມວນຜົນຄຳສັ່ງ INSERT

ໃນການສຶກສາຄວາມໄວໃນການປະມວນຜົນຂອງຄຳສັ່ງ INSERT ລະຫວ່າງພາສາ Go ແລະ Typescript ໃນ Web Application ໄດ້ດຳເນີນການທົດລອງໃນຈຳນວນແຖວ (rows) ທີ່ແຕກຕ່າງກັນແຕ່ 10 ຫາ 100,000 ແຖວ ໂດຍໄດ້ຄ່າເວລາປະມວນຜົນ (response time) ດັ່ງຕາຕະລາງລຸ່ມນີ້:

ຕາຕະລາງ 1. ບັນທຶກເວລາທີ່ໃຊ້ສຳລັບຄຳສັ່ງ INSERT

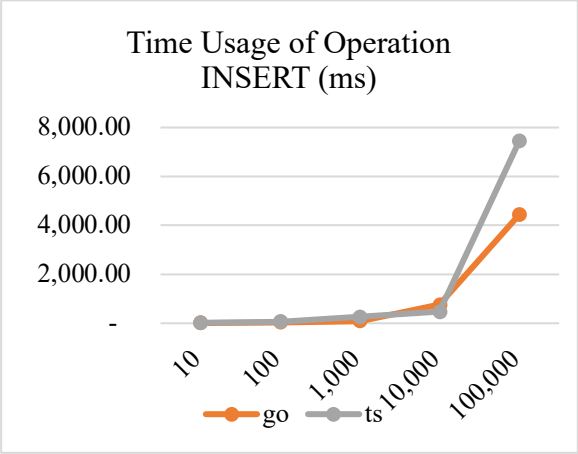
	go	**ts**
10	444.08	72.80
100	606.92	65.20
1,000	516.84	66.00
10,000	549.88	127.80
100,000	739.71	910.20

ຈາກຕາຕະລາງຂ້າງເທິງ ຜູ້ຄົ້ນຄວ້າສາມາດວິເຄາະຜົນການປະມວນຜົນຂອງການໃຊ້ຄຳສັ່ງ INSERT ໄດ້ດັ່ງນີ້ ເມື່ອມີການເພີ່ມຂະໜາດຂໍ້ມູນລະດັບຕ່ຳ ເລີ່ມແຕ່ 10-1,000 ແຖວ Typescript ສາມາດປະມວນຜົນໄດ້ໄວກວ່າ Go ຢ່າງຊັດເຈນ ຕົວຢ່າງເຊັ່ນ: ເມື່ອ INSERT ຂໍ້ມູນຈຳນວນ 100 ແຖວ, Typescript ໃຊ້ເວລາພຽງ 65.20 ms, ແຕ່ Go ໃຊ້ເຖິງ 606.92 ms, ໃນເວລາເພີ່ມຂະໜາດຂໍ້ມູນລະດັບກາງ 10,000 ແຖວ ພາສາໂປຣແກຣມ Go ແລະ Typescript ເລີ່ມມີຜົນແຕກຕ່າງກັນ ແຕ່ Go ເລີ່ມຫຼຸດປະສິດທິພາບ, ນອກຈາກນີ້ເມື່ອມີການເພີ່ມຂະໜາດຂໍ້ມູນລະດັບໃຫຍ່ຂຶ້ນເປັນ 100,000 ແຖວ Go ໃຊ້ເວລາປະມວນຜົນພຽງ 739.71 ms, ໃນຂະນະທີ່ Typescript ໃຊ້ເຖິງ 910.20 ms

ດັ່ງນັ້ນຈຶ່ງສະຫຼຸບໄດ້ວ່າ Go ມີ scalability ສູງ ແລະ ຈັດການ workload ໃຫຍ່ໄດ້ດີ ຜົນການທົດລອງສະທ້ອນໃຫ້ເຫັນວ່າ: TypeScript ເໝາະກັບງານເລັກ ແລະ ການປະມວນຜົນທີ່ບໍ່ຊັບຊ້ອນ ສ່ວນ Go ເໝາະກັບງານທີ່ມີຂໍ້ມູນຈຳນວນຫຼາຍ ຫຼື ຕ້ອງການປະສິດທິພາບສູງຢ່າງຕໍ່ເນື່ອງ.

ຮູບທີ 3: ຜົນການດຳເນີນງານ INSERT ຂໍ້ມູນ

ດັ່ງນັ້ນ, ໃນການສ້າງ Web Application ທີ່ຈະມີການ INSERT ຂໍ້ມູນຈຳນວນຫຼາຍ (ເຊັ່ນລະບົບບັນທຶກລາຍການ, Big Data, ຫຼື batch processing), Go ແມ່ນຕົວເລືອກທີ່ເໝາະສົມ ແລະ ມີຄວາມຍືດຫຍຸ່ນໃນການຮອງຮັບການໃຊ້ງານຂະໜາດໃຫຍ່ ເຊິ່ງສະແດງອອກໃນເສັ້ນສະແດງດັ່ງລຸ່ມນີ້:



ຮູບທີ 4 ເສັ້ນສະແດງປຽບທຽບເວລາການໃຊ້ INSERT

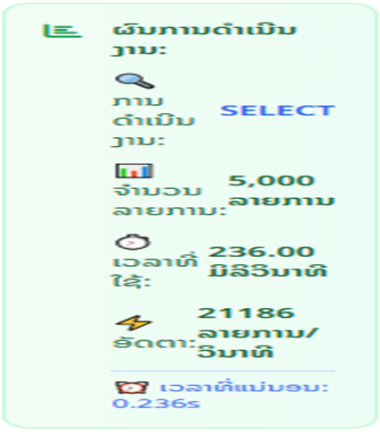
4.2 ຄວາມໄວໃນການປະມວນຜົນຄຳສັ່ງ SELECT

ໃນການສຶກສາຄວາມໄວໃນການປະມວນຜົນຂອງຄຳສັ່ງ SELECT ລະຫວ່າງພາສາ Go ແລະ Typescript ໄດ້ມີການທົດລອງຕາມຂະໜາດຂອງຂໍ້ມູນທີ່ແຕກຕ່າງກັນ ຄື 10, 100, 1,000, 10,000 ແລະ 100,000 ແຖວ. ຜົນການທົດລອງໄດ້ສະແດງໃນຕາຕະລາງດັ່ງລຸ່ມນີ້:

ຕາຕະລາງ 2. ບັນທຶກເວລາທີ່ໃຊ້ສຳລັບຄຳສັ່ງ SELECT

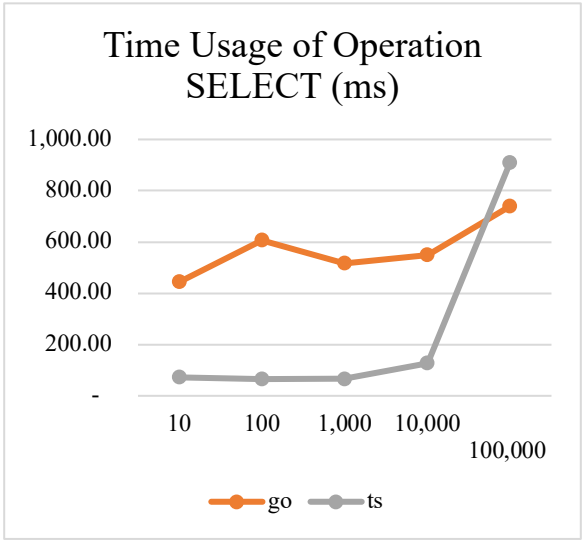
Rows	go	ts
10	444.08	72.80
100	606.92	65.20
1,000	516.84	66.00
10,000	549.88	127.80
100,000	739.71	910.20

ຈາກຕາຕະລາງຂ້າງເທິງ ຜູ້ຄົ້ນຄວ້າສາມາດວິເຄາະຜົນການປະມວນຜົນຂອງການໃຊ້ຄຳສັ່ງ SELECT ໄດ້ດັ່ງນີ້ ເມື່ອມີການເພີ່ມຂະໜາດຂໍ້ມູນລະດັບຕ່ຳ ເລີ່ມແຕ່ 10-1,000 ແຖວ Typescript ສາມາດປະມວນຜົນໄດ້ໄວກວ່າ Go ຢ່າງຊັດເຈນ ຕົວຢ່າງ: 10 ແຖວຂຶ້ນໄປ Go ໃຊ້ເວລາໃນການປະມວນຜົນ 444.08 ms, ແຕ່ Typescript ໃຊ້ເວລາໃນການປະມວນຜົນພຽງ 72.80 ms ແລະ ເມື່ອເພີ່ມຈຳຂໍ້ມູນ 100 ແຖວຂຶ້ນໄປ Go ໃຊ້ເວລາໃນການປະມວນຜົນ 606.92 ms ແຕ່ Typescript ໃຊ້ເວລາໃນການປະມວນຜົນພຽງ 65.20 ms; ໃນເວລາເພີ່ມຂະໜາດຂໍ້ມູນ 10,000 ແຖວ, Typescript ມີຄວາມໄວກວ່າ Go ຢ່າງຊັດເຈນ ຕົວຢ່າງ: Go ໃຊ້ເວລາ 549.88 ms ແຕ່ Typescript ໃຊ້ເວລາພຽງ 127.80 ms, ສະທ້ອນໃຫ້ເຫັນວ່າ Typescript ຍັງສາມາດຮັບມືກັບຂໍ້ມູນຂະໜາດປານກາງໄດ້ໄວກວ່າ Go.



ຮູບທີ 5: ຜົນການດຳເນີນງານ SELECT ຂໍ້ມູນ

ແຕ່ເມື່ອເວລາເຮົາຈຳນວນຂໍ້ມູນຂຶ້ນ 100,000 ແຖວ Go ສາມາດຄວບຄຸມເວລາໄດ້ດີກວ່າ Typescript Go: 739.71 ms, TS: 910.20 ms. ຜົນການປຽບທຽບສະແດງໃຫ້ເຫັນວ່າ Typescript ເໝາະສົມກັບການດຶງຂໍ້ມູນຂະໜາດນ້ອຍ ແລະ ປະມວນຜົນໄດ້ໄວກວ່າໃນໄລຍະສັ້ນແຕ່ເມື່ອຈຳນວນຂໍ້ມູນເພີ່ມຂຶ້ນ ປະສິດທິພາບຂອງ Go ຈະແມ່ນທີ່ໂດດເດັ່ນ ເພາະສາມາດຄວບຄຸມການໃຊ້ຊັບພະຍາກອນ ແລະ ເວລາໄດ້ດີກວ່າ ເຊິ່ງສະແດງອອກໃນເສັ້ນສະແດງລຸ່ມນີ້:



ຮູບທີ 6: ເສັ້ນສະແດງປຽບທຽບເວລາການໃຊ້ SELECT

4.3 ຄວາມໄວໃນການປະມວນຜົນຄຳສັ່ງ UPDATE

ໃນການສຶກສາຄວາມໄວໃນການປະມວນຜົນຂອງຄຳສັ່ງ UPDATE ລະຫວ່າງພາສາ Go ແລະ Typescript ໄດ້ມີການທົດລອງຕາມຂະໜາດຂອງຂໍ້ມູນທີ່ແຕກຕ່າງກັນ ຄື 10, 100, 1,000, 10,000 ແລະ 100,000 ແຖວ. ຜົນການທົດລອງໄດ້ສະແດງໃນຕາຕະລາງດັ່ງລຸ່ມນີ້:

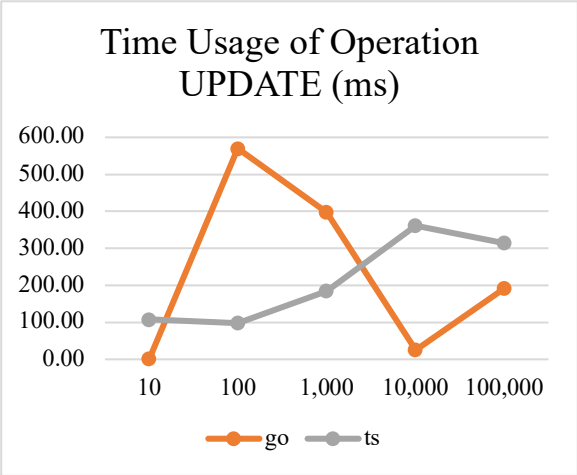
ຕາຕະລາງ 3. ບັນທຶກເວລາທີ່ໃຊ້ສຳລັບຄຳສັ່ງ UPDATE

Rows	go	ts
10	1.20	107.60
100	570.07	97.80
1,000	397.49	185.00
10,000	25.92	361.20
100,000	192.07	314.13

ຈາກຕາຕະລາງຂ້າງເທິງ ຜູ້ຄົ້ນຄວ້າສາມາດວິເຄາະຜົນການປະມວນຜົນຂອງການໃຊ້ຄຳສັ່ງ UPDATE ສຳລັບຂໍ້ມູນຈຳນວນນ້ອຍ ເລີ່ມຈາກ ຈຳນວນ 10 ແຖວ ພາສາ Go ໃຊ້ເວລາພຽງ 1.20 ms ເທົ່ານັ້ນ ຂະນະທີ່ Typescript ໃຊ້ເວລາ 107.60 ms ສະທ້ອນໃຫ້ເຫັນຄວາມໄວທີ່ແຕກຕ່າງກັນ ເກືອບເຖິງ 10 ເທົ່າ ໃນເວລາເພີ່ມຂະໜາດ ຈຳນວນ 100 ແຖວ ແລະ 1000 ແຖວ Typescript ກັບມີໜ້ອຍກວ່າ Go (100 = 97.80 ms vs 570.07 ms ແລະ 1000 = 185.00 ms vs 397.49 ms) ອາດເກີດຈາກການ loading ຫຼື batch update ທີ່ບໍ່ສົມບູນ ສ່ວນໃນຂະໜາດຂໍ້ມູນທີ່ໃຫຍ່ຂຶ້ນ (10,000 ແລະ 100,000 ແຖວ) ເຫັນໄດ້ຊັດວ່າ Go ມີຄວາມໄວກວ່າ Typescript ຢ່າງຊັດເຈນ: 10,000 rows: Go 25.92 ms vs TS 361.20 ms 100,000 rows: Go 192.07 ms vs TS 314.13 ms ສະຫຼຸບໄດ້ວ່າ ພາສາ Go ແມ່ນສາມາດປະມວນຜົນຄຳສັ່ງ UPDATE ໄດ້ໄວກວ່າ Typescript ໃນຂະໜາດຂໍ້ມູນໃຫຍ່.

ຮູບທີ 7: ຜົນການດຳເນີນງານ update ຂໍ້ມູນ

ແມ່ນພິສູດໃຫ້ເຫັນວ່າ Go ມີຄວາມເໝາະສົມສູງສຳລັບງານ backend ທີ່ຕ້ອງຈັດການຂໍ້ມູນຈຳນວນຫຼາຍ ຢ່າງມີປະສິດທິພາບ ເຊິ່ງມັນສະແດງອອກໃນເສັ້ນສະແດງລຸ່ມນີ້:



ຮູບທີ 8: ເສັ້ນສະແດງປຽບທຽບເວລາການໃຊ້ UPDATE

4.4 ຄວາມໄວໃນການປະມວນຜົນຄຳສັ່ງ DELETE

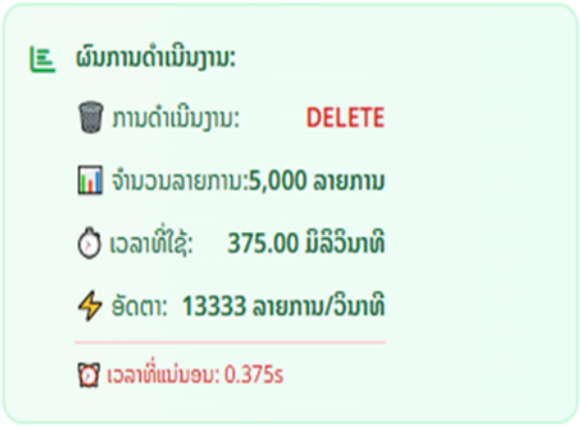
ໃນການສຶກສາຄວາມໄວໃນການປະມວນຜົນຂອງຄຳສັ່ງ DELETE ລະຫວ່າງພາສາ Go ແລະ Typescript ໄດ້ມີການທົດລອງຕາມຂະໜາດຂອງຂໍ້ມູນທີ່ແຕກຕ່າງກັນ ຄື 10, 100, 1,000, 10,000 ແລະ 100,000 ແຖວ. ຜົນການທົດລອງໄດ້ສະແດງໃນຕາຕະລາງດັ່ງລຸ່ມນີ້:

ຕາຕະລາງ 4. ບັນທຶກເວລາທີ່ໃຊ້ສຳລັບຄຳສັ່ງ DELETE

Rows	go	Ts
10	418.95	385.14
100	369.10	571.82
1,000	429.99	197.24
10,000	763.73	475.03
100,000	672.00	1,196.60

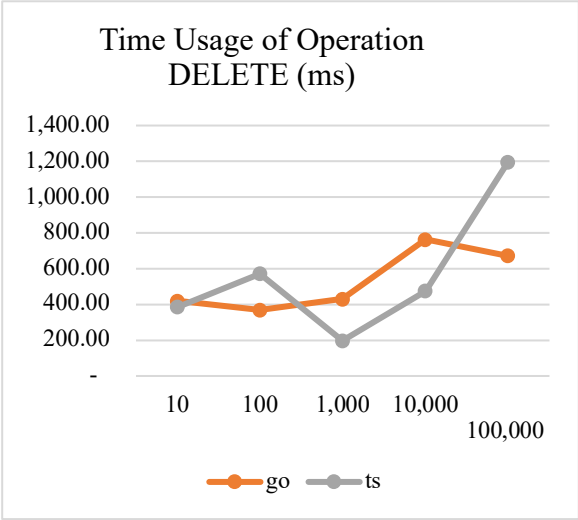
ຈາກຕາຕະລາງຂ້າງເທິງ ຜູ້ຄົ້ນຄວ້າສາມາດວິເຄາະຄວາມໄວໃນການປະມວນຜົນຄຳສັ່ງ DELETE ຂອງ Go ແລະ Typescript ໃນຈຳນວນແຖວຂໍ້ມູນທີ່ແຕກຕ່າງກັນໄດ້ດັ່ງນີ້: ເມື່ອຈຳນວນແຖວນ້ອຍ (10 ແຖວ):Typescript (385.14 ms) ສະແດງໃຫ້ເຫັນປະສິດທິພາບດີກວ່າ Go (418.95 ms) ເລັກນ້ອຍ. ນີ້ອາດຈະເປັນຍ້ອນ Overhead ເລີ່ມຕົ້ນຂອງ Go ຫຼື ຄວາມແຕກຕ່າງໃນການຈັດການ Connection ຖານຂໍ້ມູນສຳລັບການດຳເນີນງານນ້ອຍໆ. ເມື່ອເພີ່ມຈຳນວນແຖວປານກາງ (100 ແຖວ): Go (369.10 ms) ເລີ່ມຕົ້ນສະແດງປະສິດທິພາບທີ່ດີກວ່າ Typescript (571.82 ms). ນີ້ສະແດງໃຫ້ເຫັນວ່າເມື່ອ Workload ເພີ່ມຂຶ້ນເລັກນ້ອຍ, Go ສາມາດຈັດການໄດ້ຢ່າງມີປະສິດທິພາບຫຼາຍຂຶ້ນ.ເມື່ອຈຳນວນແຖວເພີ່ມຂຶ້ນອີກ 1,000 ແຖວ:Typescript (197.24 ms) ມີປະສິດທິພາບດີກວ່າ Go (429.99 ms) ຢ່າງເຫັນໄດ້ຊັດໃນຈຸດນີ້.

ນີ້ເປັນຜົນທີ່ໜ້າສົນໃຈ ແລະ ອາດຈະເກີດຈາກການຈັດການ Transaction/Batching: Typescript ອາດຈະມີການຈັດການ Batch DELETE ທີ່ມີປະສິດທິພາບກວ່າສຳລັບຈຳນວນແຖວນີ້, ຫຼື ORM/Driver ທີ່ໃຊ້ມີການOptimization ສະເພາະ. ການປັບປຸງ Driver/Runtime: V8 Engine ຂອງ Node.js ອາດມີການ Optimization ສະເພາະສຳລັບການດຳເນີນງານ I/O ບາງປະເພດໃນຊ່ວງຂໍ້ມູນຂະໜາດນີ້. ເມື່ອຈຳນວນແຖວຫຼາຍ (10,000 ແຖວ): Typescript (475.03 ms) ຍັງຄົງມີປະສິດທິພາບດີກວ່າ Go (763.73 ms). ນີ້ຊີ້ໃຫ້ເຫັນວ່າການ Optimization ໃດໆທີ່ Typescript ມີສຳລັບ 1,000 ແຖວນັ້ນຍັງຄົງມີຜົນຢູ່, ຫຼື Go ອາດຈະມີ Overhead ເພີ່ມເຕີມເມື່ອຈຳນວນແຖວເລີ່ມເພີ່ມຂຶ້ນແຕ່ຍັງບໍ່ທັນເຖິງຈຸດທີ່ Concurrency ຂອງມັນໂດດເດັ່ນເຕັມທີ່. ຈຳນວນແຖວຫຼາຍສຸດ (100,000 ແຖວ): Go (672.00 ms) ກັບມາສະແດງປະສິດທິພາບທີ່ດີກວ່າ Typescript (1,196.60 ms) ຢ່າງຊັດເຈນ.



ຮູບທີ 9: ຜົນການດຳເນີນງານ DELETE ຂໍ້ມູນ

ນີ້ເປັນສິ່ງທີ່ຄາດເດົາໄດ້ຫຼາຍຂຶ້ນ ເພາະເມື່ອ Workload ໃຫຍ່ຂຶ້ນ, ຄວາມສາມາດຂອງ Go ໃນການຈັດການ Concurrency ແລະ Memory Efficiency ຈະກາຍເປັນປັດໄຈທີ່ສຳຄັນ. Typescript ອາດຈະປະສົບບັນຫາ Blocking Event Loop ຫຼື Memory Overhead ເມື່ອປະມວນຜົນການລຶບຂໍ້ມູນຈຳນວນຫຼວງຫຼາຍ ເຊິ່ງມັນສະແດງອອກໃນເສັ້ນສະແດງດັ່ງລຸ່ມນີ້:



ຮູບທີ 10: ເສັ້ນສະແດງປຽບທຽບເວລາການໃຊ້ DELETE

5. ອະພິປາຍຜົນ

ຜົນການທົດລອງສະແດງໃຫ້ເຫັນວ່າ Go ມີຄວາມໂດດເດັ່ນໃນດ້ານ “ຄວາມໄວໃນການປະມວນຜົນ” ແລະ “ການຈັດການຂໍ້ມູນແບບຊ້ອນກັນ (concurrent)” ອີງຕາມບົດວິໄຈຂອງ ທ່ານ Lee & Lim (2022) ແລະ ທ່ານ Rodriguez et al. (2022) ທີ່ເຊື່ອວ່າ Go ເໝາະສົມກັບການນຳໃຊ້ໃນ Microservices. ໃນຂະນະທີ່ Typescript ອາດຈະບໍ່ໄວທັນ Go ແຕ່ມີຄວາມໄວໃນການພັດທະນາ ແລະ ດູແລລະບົບໄດ້ງ່ຍ ຕາມບົດວິໄຈຂອງ ທ່ານ Feigenbaum (2021) ແລະ ທ່ານ Thompson & Davis (2023). ນອກຈາກນັ້ນ ທ່ານ Ardiansyah & Fatwanto (2022) ຍັງພົບວ່າ Go ໃຊ້ໜ່ວຍຄວາມຈຳນ້ອຍກວ່າ Javascript. ຜົນການທົດລອງຊີ້ໃຫ້ເຫັນວ່າ Go ໄດ້ໄປໃນທາງ native thread execution ຈຶ່ງກະທົບໃຫ້ເກີດຄວາມໄວ ເທົ່າທີ່ Typescript ໄດ້ໃຊ້ Node.js ທີ່ມີອີເວັນລູບເປັນພື້ນຖານ ຈຶ່ງເປັນປັດໄຈສຳຄັນຕໍ່ການເລືອກໃຊ້ພາສາໃນການພັດທະນາ Web Application. ເຖິງຢ່າງໃດກໍຕາມ Typescript ກໍ່ຍັງເໝາະສົມໃນສະຖານະການທີ່ຕ້ອງການພັດທະນາແບບ rapid prototyping ຫຼື ໃນທີມນັກພັດທະນາ ທີ່ມີພື້ນຖານ JavaScript ຢູ່ແລ້ວ ເຊິ່ງຜົນຈາກການທົດລອງ ແລະ ບົດວິໄຈຊີ້ແຈງວ່າ: ຖ້າເນັ້ນໃສ່ຄວາມໄວ ແລະ ການປະມວນຜົນແບບ concurrent request Go ແມ່ນເໝາະສົມກວ່າ ແຕ່ຖ້າຫາກເນັ້ນໃສ່ຄວາມໄວໃນການຂຽນໂຄດ ແລະ ຮູບແບບການໃຊ້ງານດູແລລະບົບແບບຕໍ່ເນື່ອງ Typescript ແມ່ນມີປະສິດທິພາບດີກວ່າ.

6. ສະຫຼຸບຜົນ

ໃນການປຽບທຽບຄວາມໄວໃນການປະມວນຜົນຄຳສັ່ງ SQL ລະຫວ່າງ ພາສາ Go ແລະ ພາສາ Typescript, ຜົນການທົດລອງພົບວ່າ ແຕ່ລະພາສາມີຈຸດແຂງ ແລະ ຈຸດອ່ອນທີ່ແຕກຕ່າງ

ກັນ ຕາມຂະໜາດຂອງຂໍ້ມູນ ແລະ ປະເພດຂອງຄຳສັ່ງ. ໃນຄຳສັ່ງ INSERT ແລະ DELETE, ພາສາ Typescript ສາມາດ ປະມວນຜົນໄດ້ໄວກວ່າໃນຂໍ້ມູນທີ່ນ້ອຍ ແລະ ຂະໜາດກາງ ແຕ່ ເມື່ອຂະໜາດຂອງຂໍ້ມູນເພີ່ມຂຶ້ນ ພາສາ Go ຈະເລີ່ມສະແດງ ປະສິດທິພາບທີ່ດີກວ່າຢ່າງຊັດເຈນ. ສ່ວນໃນຄຳສັ່ງ SELECT ແລະ UPDATE, ພາສາ Go ສາມາດປະມວນຜົນໄດ້ໄວກວ່າ Typescript ໃນທຸກຂະໜາດຂອງຂໍ້ມູນ. ຈາກຜົນການທົດລອງນີ້ ສາມາດສະຫຼຸບໄດ້ວ່າ ພາສາ Go ເໝາະສົມສຳລັບການໃຊ້ງານທີ່ ຕ້ອງປະມວນຂໍ້ມູນຈຳນວນຫຼາຍ ແລະຕ້ອງການຄວາມໄວສູງ ໃນ ຂະນະທີ່ Typescript ເໝາະສຳລັບງານຂະໜາດນ້ອຍ ຫຼື ການ ພັດທະນາທີ່ຄ້ອນຂ້າງໄວ. ດັ່ງນັ້ນ ການເລືອກໃຊ້ພາສາໃດຄວນ ພິຈາລະນາຕາມເປົ້າໝາຍ ແລະ ລັກສະນະຂອງແອັບພລິເຄຊັນທີ່ ຈະພັດທະນາ.

7. ຂໍ້ສະເໜີແນະ

ໃນການທົດລອງ ແລະ ເຮັດຄົ້ນຄວ້າວິໄຈ ສາມາດເຫັນ ໄດ້ວ່າ ພາສາ Go ມີຄວາມເໝາະສົມ ການປະມວນຜົນທີ່ມີຂະ ໜາດໃຫຍ່ ແລະ ມີປະສິດທິພາບດ້ານຄວາມໄວສູງຫຼາຍ ໃນການ ຈັດການຂໍ້ມູນຈຳນວນຫຼາຍ ໂດຍສະເພາະໃນເວລາ SELECT, UPDATE, ແລະ DELETE ໃນ batch ໃຫຍ່. ດັ່ງນັ້ນແນະນຳ ໃຫ້ນຳໄປໃຊ້ໃນລະບົບ backend ທີ່ຈຳເປັນຕ້ອງການຄວາມໄວ ແລະ ຄວາມສະຖຽນຂອງການປະມວນຜົນ. ສຳລັບ Typescript ແມ່ນເໝັ້ນໃນການພັດທະນາໄວ ຫຼື ຂໍ້ມູນນ້ອຍ ໃນຄຳສັ່ງ INSERT ແລະ DELETE ທີ່ມີຂໍ້ມູນບໍ່ຫຼາຍ, ສາມາດປະມວນ ຜົນໄດ້ໄວ ແລະ ພັດທະນາແອັບພລິເຄຊັນໄດ້ໄວ ຈຶ່ງເໝາະກັບງານ ທີ່ເນັ້ນການພັດທະນາ ລະບົບຂະໜາດນ້ອຍ.

8. ຂໍ້ຈຳກັດຂອງການຄົ້ນຄວ້າ

ຂ້າພະເຈົ້າໃນນາມຜູ້ຄົ້ນຄວ້າວິທະຍາສາດ ຂໍປະຕິຍານ ຕົນວ່າ ຂໍ້ມູນທັງໝົດທີ່ມີໃນບົດຄວາມວິຊາການດັ່ງກ່າວນີ້ ແມ່ນບໍ່ ມີຂໍ້ຂັດແຍ່ງທາງຜົນປະໂຫຍດກັບພາກສ່ວນໃດ ແລະ ບໍ່ໄດ້ເອື້ອ ປະໂຫຍດໃຫ້ກັບພາກສ່ວນໃດພາກສ່ວນໜຶ່ງ, ກໍລະນີມີການ ລະເມີດໃນຮູບການໃດໜຶ່ງ ຂ້າພະເຈົ້າມີຄວາມຍິນດີ ທີ່ຈະ ຮັບຜິດຊອບແຕ່ພຽງຜູ້ດຽວ.

9. ເອກະສານອ້າງອີງ

Costanza, P., Herzeel, C., & Verachtert, W. (2019). Comparing ease of programming in C++, go, and java for implementing a Next-Generation sequencing tool. *Evolutionary Bioinformatics*, *15*, 1176934319869015.

Lee, Y., & Lim, Y. H. (2022). Concurrency processing comparison of large data list using GO language. The Journal of the Convergence on Culture Technology, 8(2), 361-366.

Sabrina, N. K. D., Pramana, D., & Kusuma, T. M. (2023). Implementation of Golang and ReactJS in the COVID-19 Vaccination Reservation System. ADI Journal on Recent Innovation, 5(1), 1-12.

Ardiansyah, H., & Fatwanto, A. (2022). Comparison of Memory usage between REST API in Javascript and Golang. *MATRIK: Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, *22*(1), 229-240.

Valkov, I., Chechina, N., & Trinder, P. (2018, April). Comparing languages for engineering server software: erlang, go, and scala with akka. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing* (pp. 218-225).

Schmager, F., Cameron, N., & Noble, J. (2010). Gohotdraw: Evaluating the go programming language with design patterns. In *Evaluation and usability of programming languages and tools*

Kang, H., & Won, Y. (2024). GoAsap: A Proposal for a Golang New Version Detection and Analysis System from a Static Analysis Perspective. *Journal of the Korea Institute of Information Security & Cryptology*, *34*(4), 707-724.

Feigenbaum, Ph. D, B. (2021). A Brief Look at Go vs. Java. In *Go for Java Programmers: Learn the Google Go Programming Language* (pp. 5-14). Berkeley, CA: Apress.

Robu, E. (2023). Enhancing data security and protection in marketing: a comparative analysis of Golang and PHP approaches. *EcoSoEn*, *1*(3-4), 19-30.

Putra, M. D. A., Dewi, D. A., Putri, W. T. H., & Achsan, H. T. Y. (2024). Efficient Web Mining on MyAnimeList: A Concurrency-Driven Approach Using the Go Programming Language. *Journal of Applied Data Sciences*, *5*(3), 1472-1481.

Gowda, P. G. A. N. Typescript vs. JavaScript: A Comparative Analysis.

Golec, M., & Plechawska-Wójcik, M. (2022). Comparative analysis of frameworks using Typescript to build server applications. *Journal of Computer Sciences Institute*, *23*, 128-134.

Lestrelin, G., Vigiak, O., Pelletreau, A., Keohavong, B. and Valentin, C. (2012). Challenging established narratives on soil erosion and shifting cultivation in Laos. *Natural Resources Forum, 36*: 63–75.

Anderson, J., & Smith, R. (2021). Performance comparison between Java and Python in web-based systems.

Journal of Software Engineering, 15(3), 78-92.

Brown, M., & Wilson, K. (2023). Modern programming languages for web development: A comprehensive review. *International Conference on Web Technologies*, 45-58.

Chen, L., & Johnson, P. (2023). The impact of programming language choice on web application performance. *ACM Transactions on Web Technologies*, 8(2), 1-18.

Kim, S., Lee, H., & Park, J. (2020). Node.js vs PHP: A performance analysis for REST API development. *Web Development Quarterly*, 12(4), 156-170.

Lee, D., & Park, S. (2022). Concurrent request handling in modern programming languages. *Software Performance Journal*, 7(1), 23-35.

Martinez, A., Garcia, C., & Lopez, E. (2022). Cost implications of programming language selection in enterprise web applications. *Business Technology Review*, 29(8), 112-128.

Rodriguez, M., Thompson, B., & Clark, A. (2022). GO programming language performance analysis in microservices architecture. *Distributed Systems Conference Proceedings*, 234-247.

Thompson, R., & Davis, L. (2023). Typescript adoption in large-scale web applications: Benefits and challenges. *Modern Web Development*, 18(6), 89-104.

Wang, X., Zhang, Y., & Liu, Q. (2023). Memory efficiency analysis of programming languages in cloud computing environments. *Cloud Computing Research*, 11(2), 67-81.