

## 1. Decisions

When I started, 2 containers were obvious to create: MongoDB and VueJS front-end containers. To create a microservice architecture, I came up with 3 microservices for the backend:

- Auth-server: Used only for user-related operations. Consists of login and register services.
- Expense-server: Used for creating and getting expenses.
- Expense-group-server: User for expense group-related operations.

All three of these servers are connected to the MongoDB container.

This way other services can keep working even if one of the services is down.

For example, the user can still log in and create expense groups if the expense-server is down.

## 2. Challenges

- Error connecting to MongoDB: I had a problem connecting back-end servers to the MongoDB container. This caused an error in the back-end servers. I solved this problem by changing the MongoDB connection string with environment variables in the back-end servers.
- Port Problem: I had a problem where the ports were conflicting because I was using the same outside port for all back-end servers. I solved this problem by assigning different ports to each server.

## 3. Deployment Workflow

- Docker compose reads through the Dockerfile and finds Dockerfile instructions.
- Docker builds an image from the Dockerfile.
- For MongoDB, Docker downloads the MongoDB image from Docker Hub.
- Docker creates a container from the image and runs the container assigning ports and volumes as specified in the docker-compose.yml file.

Dockerfile of back-end servers (port changes between them)

FROM node:18	Docker loads node:18 image from Docker Hub.
WORKDIR /	WORKDIR sets the working directory for the container.
COPY package*.json ./	Copies package.json and package-lock.json files to the container.
RUN npm install	Installs the dependencies
COPY . .	Copies all the files from the current directory to the container
EXPOSE 3000	Exposes port 3000
CMD ["node", "server.js"]	Runs “node server.js”

## Dockerfile of front-end application

FROM node:18	Docker loads node:18 image from Docker Hub.
WORKDIR /app	WORKDIR sets the working directory for the container.
COPY package*.json ./	Copies package.json and package-lock.json files to the container.
RUN npm install -g pnpm	Installs pnpm globally
RUN pnpm install	Installs the dependencies (using pnpm)
COPY . .	Copies all the files from the current directory to the container
EXPOSE 5173	Exposes port 5173
CMD ["pnpm", "run", "dev", "--host"]	Runs “pnpm run dev --host”

## 4. System Architecture

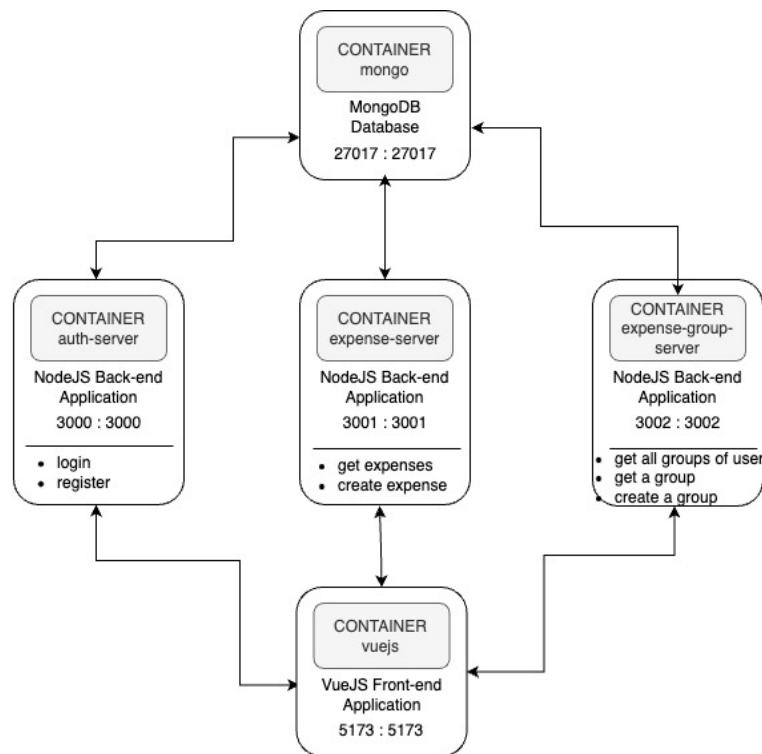


Figure 2 docker compose logs

docker compose ps						
NAME	IMAGE	COMMAND	SERVICE	CREATED	STATUS	PORTS
ceng495-hw2-auth-server-1	ceng495-hw2-auth-server	"docker-entrypoint.s..."	auth-server	29 minutes ago	Up 21 minutes	0.0.0.0:3000->3000/tcp
ceng495-hw2-expense-group-server-1	ceng495-hw2-expense-group-server	"docker-entrypoint.s..."	expense-group-server	29 minutes ago	Up 21 minutes	0.0.0.0:3002->3002/tcp
ceng495-hw2-expense-server-1	ceng495-hw2-expense-server	"docker-entrypoint.s..."	expense-server	29 minutes ago	Up 21 minutes	0.0.0.0:3001->3001/tcp
ceng495-hw2-mongo-1	mongo	"docker-entrypoint.s..."	mongo	29 minutes ago	Up 21 minutes	0.0.0.0:27017->27017/tcp
ceng495-hw2-vuejs-1	ceng495-hw2-vuejs	"docker-entrypoint.s..."	vuejs	29 minutes ago	Up 3 seconds	0.0.0.0:5173->5173/tcp

Figure 3 docker compose ps

















Name	Image	Status	CPU (%)	Port(s)
 <b>ceng495-hw2</b>		Running (5/5)	2.37%	
 <b><a href="#">mongo-1</a></b> 14fe15b35518 	<a href="#">mongo</a>	Running	2.21%	<a href="#">27017:27017</a> 
 <b><a href="#">auth-server-1</a></b> 96c995e887ff 	<a href="#">ceng495-hw2-auth-</a>	Running	0%	<a href="#">3000:3000</a> 
 <b><a href="#">expense-server-1</a></b> 82b13f20f70c 	<a href="#">ceng495-hw2-exper</a>	Running	0%	<a href="#">3001:3001</a> 
 <b><a href="#">expense-group-server-1</a></b> 2fb072d58366 	<a href="#">ceng495-hw2-exper</a>	Running	0%	<a href="#">3002:3002</a> 
 <b><a href="#">vuejs-1</a></b> 80c65fd2ad91 	<a href="#">ceng495-hw2-vuejs</a>	Running	0.16%	<a href="#">5173:5173</a> 

Figure 4 Docker Desktop