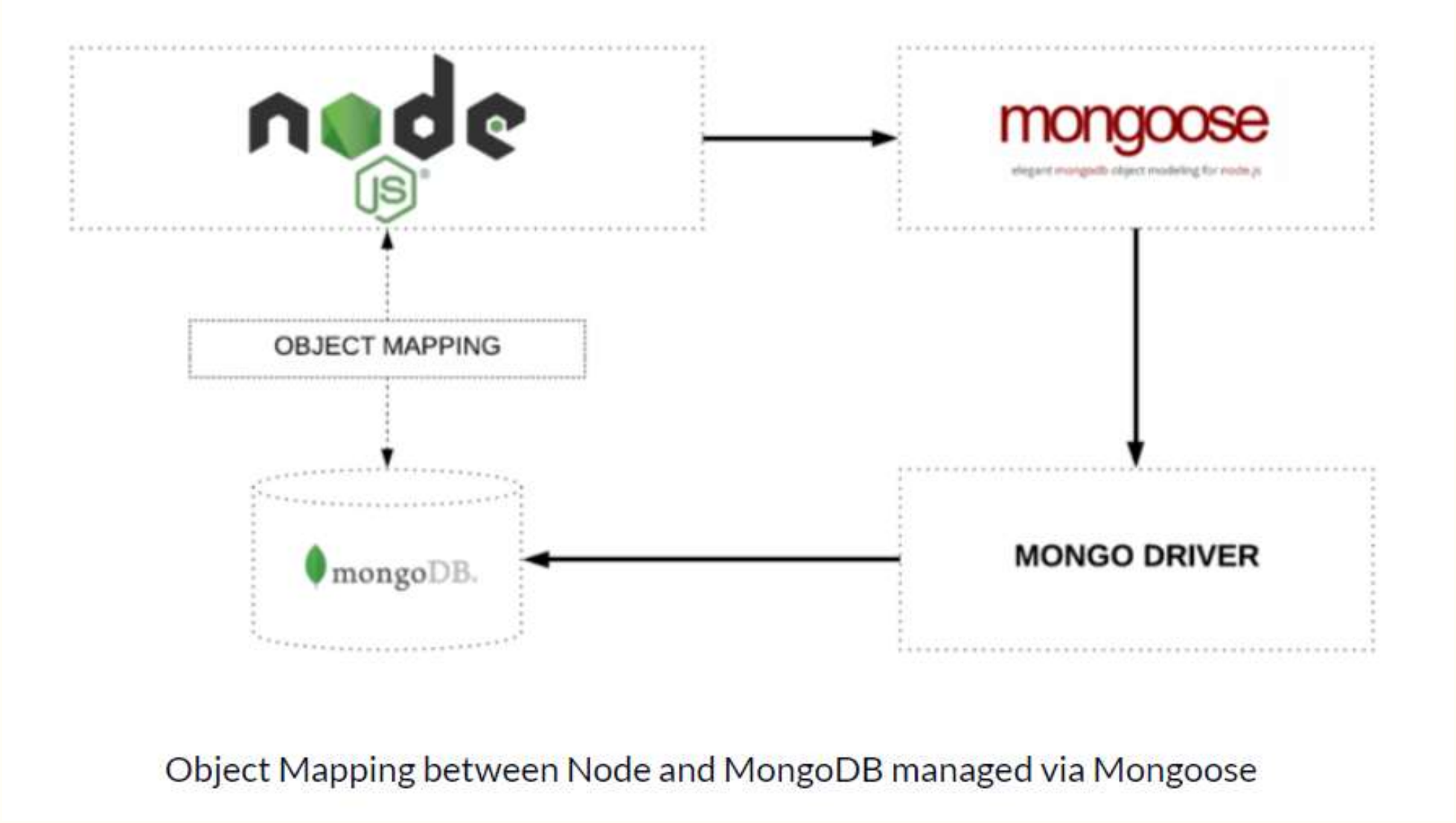
Mongoose

Qinshift Academy

WHAT IS MONGOOSE?

Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

Mongoose in Node JS project

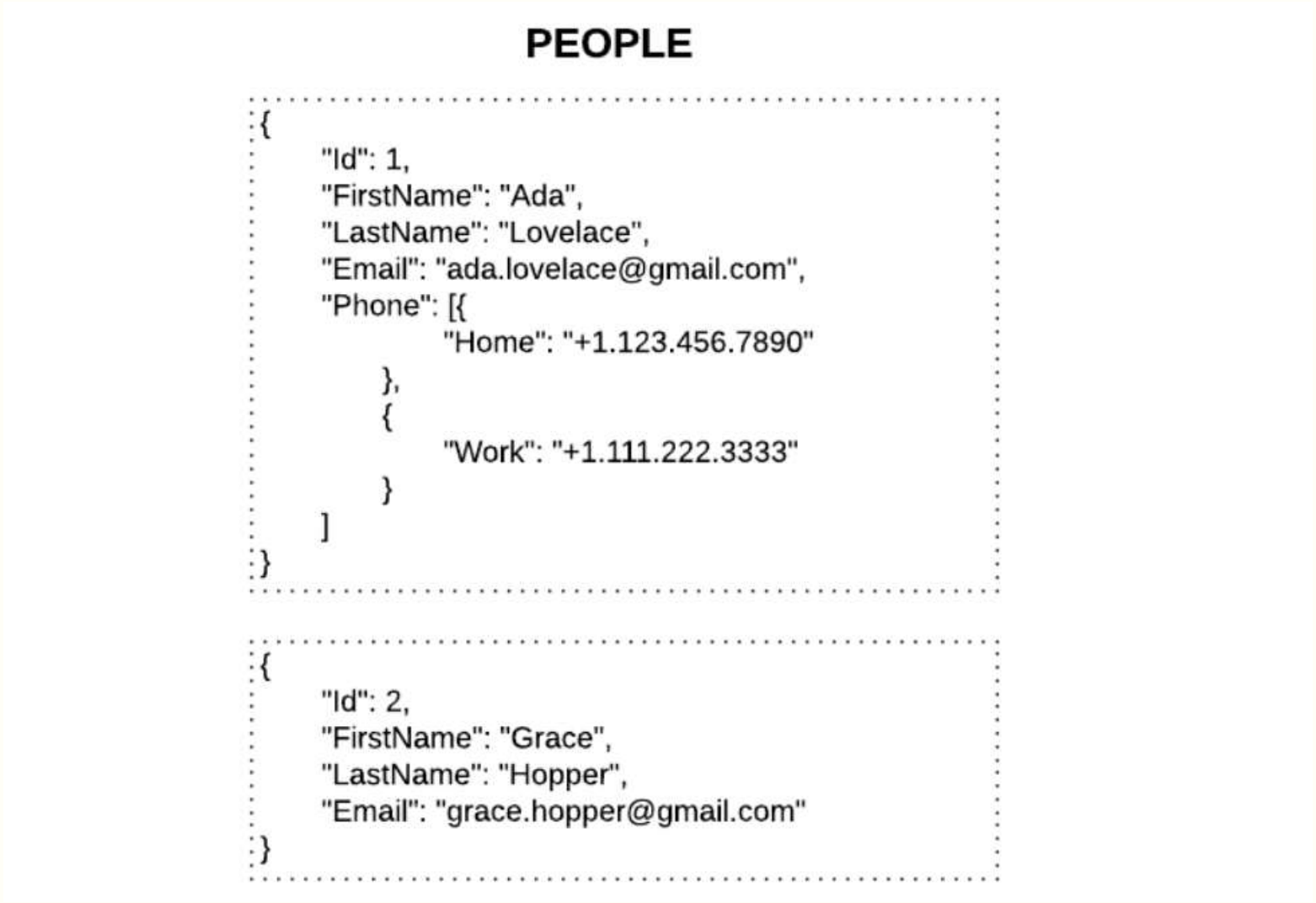


Object Mapping between Node and MongoDB managed via Mongoose

Mongo DB

MongoDB is a schema-less NoSQL document database. It means you can store JSON documents in it, and the structure of these documents can vary as it is not enforced like SQL databases. This is one of the advantages of using NoSQL as it speeds up application development and reduces the complexity of deployments.

Storing data in Mongo DB



SCHEMA / SCHEMA TYPES

- A Mongoose **schema** is a document data structure (or shape of the document) that is enforced via the application layer.
- While Mongoose schemas define the overall structure or shape of a document, **SchemaTypes** define the expected data type for individual fields (String, Number, Boolean, and so on).

You can also pass in useful options like **required** to make a field non-optional, **default** to set a default value for the field, and many more.

MODELS

Models are higher-order constructors that take a schema and create an instance of a document equivalent to records in a relational database.

```
import { Schema, model } from "mongoose";

const UserSchema = new Schema(
    {
        firstName: { type: String, required: true },
        lastName: { type: String, required: true },
        email: { type: String, required: true, unique: true },
        password: { type: String, required: true, unique: true },
    },
    { timestamps: true },
);

const User = model("users", UserSchema);

export default User;
```

MODELS

Models are higher-order constructors that take a schema and create an instance of a document equivalent to records in a relational database.

```
const puppySchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  age: Number
});

const Puppy = mongoose.model('Puppy', puppySchema);
```

Schema explained

In the code above, puppySchema defines the shape of the document which has two fields, name, and age

The SchemaType for name is String and for age is Number. Note that you can define the SchemaType for a field by using an object with a type property like with name. Or you can apply a SchemaType directly to the field like with age.

Also, notice that the SchemaType for name has the option required set to true. To use options like required and lowercase for a field, you need to use an object to set the SchemaType.

At the bottom of the snippet, puppySchema is compiled into a model named Puppy, which can then be used to construct documents in an application.

Questions?

Trainer Name

Trainer

trainer@mail.com

Assistant Name

Assistant

asistant@mail.com

Qinshift Academy