* **[Hapi](https://hapi.dev/)** framework forNode js.
* **SQLite** is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine.
* **Sequelize** is a promise-based ORM for Node.js and io.js. It supports the dialects **SQLite.**
* **Pug** is available via npm:

**MVC**

**Modal -** the part of our application that will deal with the database or any data-related functionality.

* In our case we used **Sequelize** a promise-based ORM for Nodejs, which supports the dialects **SQLite.**
* **Get Data**
* Handles data logic
* Interacts with database

**View** – everything the user will see — basically, the pages that we’re going to send to the client.

* We used **Pug**.
* Get presentation
* Handles data presentation
* Dynamically rendered

**Controller** - the logic of our site, and the glue between models and views. Here we call our models to get the data, then we put that data on our views to be sent to the users.

* Request
* Response
* Handles request flow
* Never handles data logic
* Process requests (GET, POST, PUT, DELETE)
* Receives input (from view, url)

1. **View** 
   1. a.show-modal(href='#') Publish
   2. modal will show with method=’POST’
      1. form(action='/note' method='POST').note-form#note-form
2. **Routes**
   1. {

   method: "POST",

   path: "/note",

handler: Note.create,

config: {

description: "Adds a new note"

}

},

1. **Controller**
   1. const { Note } = require("../models/");
   2. module.exports = {
   3. create: async (request, h) => {
   4. const result = await Note.create({
   5. date: new Date(),
   6. title: request.payload.noteTitle,
   7. slug: Slugify(request.payload.noteTitle, { lower: true }),
   8. description: request.payload.noteDescription,
   9. content: request.payload.noteContent
   10. });
   11. // Generate a new note with the 'result' data
   12. return Pug.renderFile(
   13. Path.join(\_\_dirname, "../views/components/note.pug"),
   14. {
   15. note: result
   16. }
   17. );
   18. },
2. **Model**
   1. module.exports = (sequelize, DataTypes) => {
   2. const Note = sequelize.define("Note", {
   3. date: {
   4. type: DataTypes.DATE,
   5. get: function() {
   6. return Moment(this.getDataValue("date")).format("MMMM Do, YYYY");
   7. }
   8. },
   9. title: DataTypes.STRING,
   10. slug: DataTypes.STRING,
   11. description: DataTypes.STRING,
   12. content: DataTypes.STRING
   13. });
   14. return Note;
   15. };
3. **Controller**
   1. read: async (request, h) => {
   2. const note = await Note.findOne({
   3. where: {
   4. slug: request.params.slug
   5. }
   6. });
   7. return h.view("note", {
   8. note,
   9. page: `${note.title}—Notes Board`,
   10. description: note.description
   11. });
   12. },
4. **View**
   1. main(container).notes-list
   2. each note in data.notes
   3. include components/note
   4. hr