

Curso Técnico Superior Profissional em: Tecnologias e Programação de Sistemas de Informação

1.º Ano/ 2.º Semestre

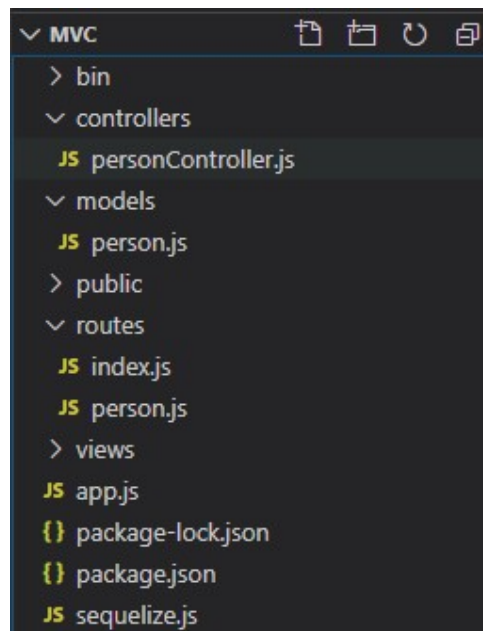
Unidade Curricular: Desenvolvimento Web - Back-End

Docente: David Jardim

FICHA DE TRABALHO 10

Exercícios:

1. Crie uma pasta para a aula 10
2. O objetivo deste exercício é replicar a ficha da aula passada utilizando o padrão de desenho de software MVC
 - a. Utilizando o **express-generator** crie um projeto chamado mvc com o **view engine ejs**
 - b. Crie uma pasta **models** para os modelos e outra **controllers** para os controllers na raiz do projeto
 - i. Adicione um ficheiro **person.js** na pasta **models** e implemente o modelo em sequelize
 - ii. Adicione um ficheiro **personController.js** à pasta **controllers**
 - c. Adicione um ficheiro **person.js** na pasta **routes** e implemente os endpoints necessários
 - d. Adicione um ficheiro **sequelize.js** na raiz do projeto e implemente toda a lógica necessária para efetuar a ligação à base de dados e sincronizar os modelos



- e. Implemente toda a lógica de CRUD à base de dados no respetivo controller

3. Faça download do projeto template disponibilizado no Moodle para efetuar autenticação de utilizadores
- Instale a extensão **TODO Tree** e inspecione as tarefas a fazer
 - Execute o comando ***npm install*** para instalar todas as dependências do projeto
 - Adicione todas as pastas e ficheiros necessários para que o projeto obedeça ao padrão de desenho de software MVC
 - Utilizando o sequelize implemente toda a lógica necessária para representar um utilizador, replique o código implementado na alínea 2
 - O modelo *Users* possui as seguintes colunas:
 - id* (int auto-increment)
 - password* (var char)
 - email* (var char)
 - Implemente um método de ***signup*** para criação de novos utilizadores via formulário WEB que serão adicionados à base de dados
 - Implemente um método de ***login*** para permitir que os utilizadores efetuem autenticação via formulário WEB com verificação na base de dados e tenham acesso a páginas restritas
 - Faça tratamento de todos os erros que poderão ocorrer na criação de utilizadores ou na autenticação de utilizadores já existentes
 - Implemente um método de autenticação *custom* (aceda à documentação em <http://www.passportjs.org/docs/authenticate/>)
 - Implemente uma nova rota com acesso protegido (users) e um novo controller (usersController) para disponibilizar toda a lógica de CRUD em forma de API

URI	Método HTTP	Body	Resultado
/users	GET	empty	Show list of all the users.
/users	POST	JSON String	Add details of new user.
/users/:id	DELETE	empty	Delete an existing user.
/users/:id	GET	empty	Show details of a user.
/users/:id	PUT	JSON String	Update details of a user

Tabela 1 - Endpoints para o recurso User