Indice

1. Scopo del Documento
2. BackEnd
    2.1. Struttura
    2.2. Dependencies
    2.3. DataBase
    2.4. APIs
        2.4.1. ResourceS Extraction from the Class Diagram
        2.4.2. Resources Model
        2.4.3. Sviluppo
        2.4.4. Documentazione API
        2.4.5. Testing Jests
        2.4.6. .env

3. FrontEnd
    3.1. Struttura
    3.2. Dependencies
    3.3. User Flow
    3.4. Sviluppo
        3.4.1. page.tsx
    3.5. Schermate

4. GitHub Repository

5. Deployment

1. Scopo del Documento

2. BackEnd

Il BackEnd è stato realizzato con il run-time enviroment `Node.js`. E visibile nella repository `G28-Antico-Granaio/D4/project/server`

2.1. Struttura

```
/G28-Antico-Granaio/D4/project/server
├──node_modules
├──src
│     ├──models
│     ├──routes
│     ├──services
│     └──index.ts
├──.env
├──.gitignore
├──package-lock.json
├──package.json
└──tsconfig.json
```

La struttura del BackEnd è suddivisa in più cartelle, in particolare:

- **`src/models`**: ciao
- **`src/routes`**: ciao
- **`src/services`**: ciao
- **`src/index`**: ciao
- **`.env`**: ciao

2.2. Dependencies

Si descrivono nello specifico le librerie più rilevanti:

- **dotenv**:
- **express**:
- **mongodb**:
- **mongoose**:

2.3. DataBase

Il BackEnd si interfaccia con un Database su MongoDB Atlas. MongoDb è un Database NoSQL, e quindi non relazionale, con struttura Document-Oriented.
Si utilizza Mongoose per interfacciarsi con il DataBase in maniera più strutturata, definendo degli *Schemas.* Il DataBase bookeasy-cluster-0 è suddiviso inn più collections, per ogniuna delle quali definiamo uno Schemas mongoose:

IMMAGINE

2.4. APIs

2.4.1. Resources Extraction from the Class Diagram

SCHEMA VERDE E GIALLO API

2.4.2. Resources Models

grafico con freccette blu e nero

2.4.3. Sviluppo

Si è utilizzato uno style di programmazione async/await e gestione degli errori con try/catch.

2.4.4. Documentazione APIs

2.4.5. Testing Jest

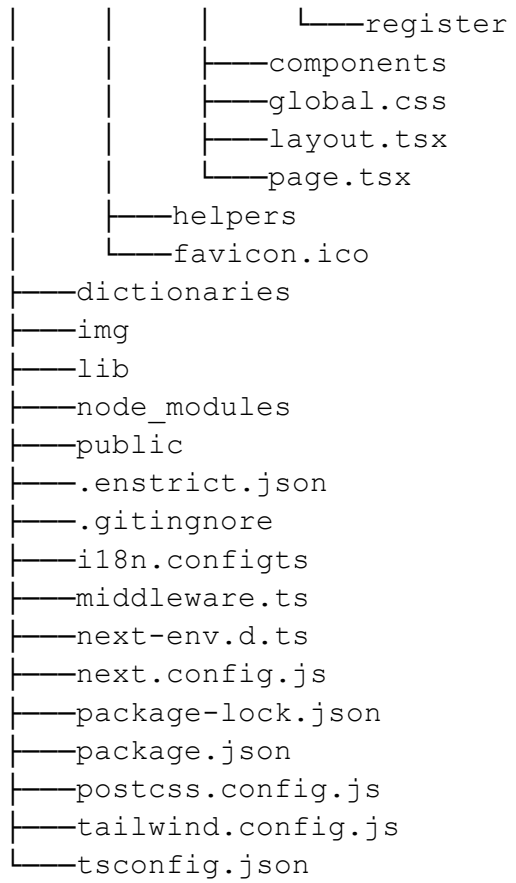
2.4.6. .env

3. FrontEnd

Il Front End è stato realizzato con Next.js 14, seguendo i suoi principi (*The React Way*). La WebApp è quindi una Single Page Applicatio. è visibile nella repository `/G28-Antico-Granaio/D4/project/app`

3.1. Struttura

```
/G28-Antico-Granaio/D4/project/app
├──.next
├──app
│     ├──[lang]
│     │     ├──(auth)
│     │     │     ├──(password)
│     │     │     │     ├──password-recovery
│     │     │     │     └──reset-password
│     │     │     ├──change-credentials
│     │     │     ├──login
```

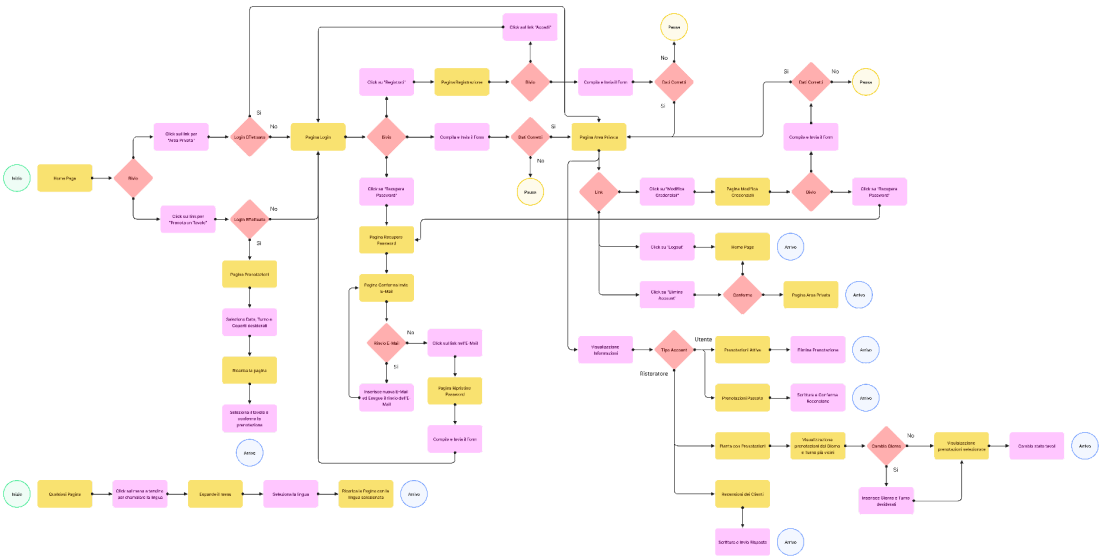


La struttura del BackEnd è suddivisa in più cartelle, in particolare:

3.2. Dependencies

```
"dependencies": {
  "@formatjs/intl-localematcher": "^0.5.2",
  "@reduxjs/toolkit": "^1.9.7",
  "antd": "^5.11.4",
  "axios": "^1.6.2",
  "bcryptjs": "^2.4.3",
  "firebase": "^10.6.0",
  "i18n": "^0.15.1",
  "jsonwebtoken": "^9.0.2",
  "mongoose": "^8.0.1",
  "negotiator": "^0.6.3",
  "next": "14.0.3",
  "react": "^18",
  "react-dom": "^18",
  "react-i18next": "^13.5.0",
  "react-redux": "^8.1.3",
  "redux": "^4.2.1"
},
"devDependencies": {
  "@types/negotiator": "^0.6.3",
  "@types/node": "^20",
  "@types/react": "^18",
  "@types/react-dom": "^18",
  "autoprefixer": "^10.0.1",
  "eslint": "^8",
  "eslint-config-next": "14.0.3",
  "postcss": "^8",
  "tailwindcss": "^3.3.0",
  "typescript": "^5"
}
```

3.3. User Flow



3.4. Sviluppo

3.5. Schermate

IMG Sito

4. GitHub Repository

5. Deployment

https://www.visual-paradigm.com/support/documents/vpuserguide/276/3420/85154_modelint.html

https://app.flowmapp.com/projects/97bcc5e2-ee19-4ef0-88d6-440c74d7de25/userflow/c92e69f1-294a-45dd-a9d0-ac56ae26a31f

Alessandro Busola - 13/12/2023 - 10.45-12.55 + 14.00