

Atsuhiko Mochizuki

PIZZAHUTTE

Création d'une application Javascript FullStack complète

Administration d'une interface clientèle WEB dédiée à une infrastructure de type Pizzeria



Projet individuel

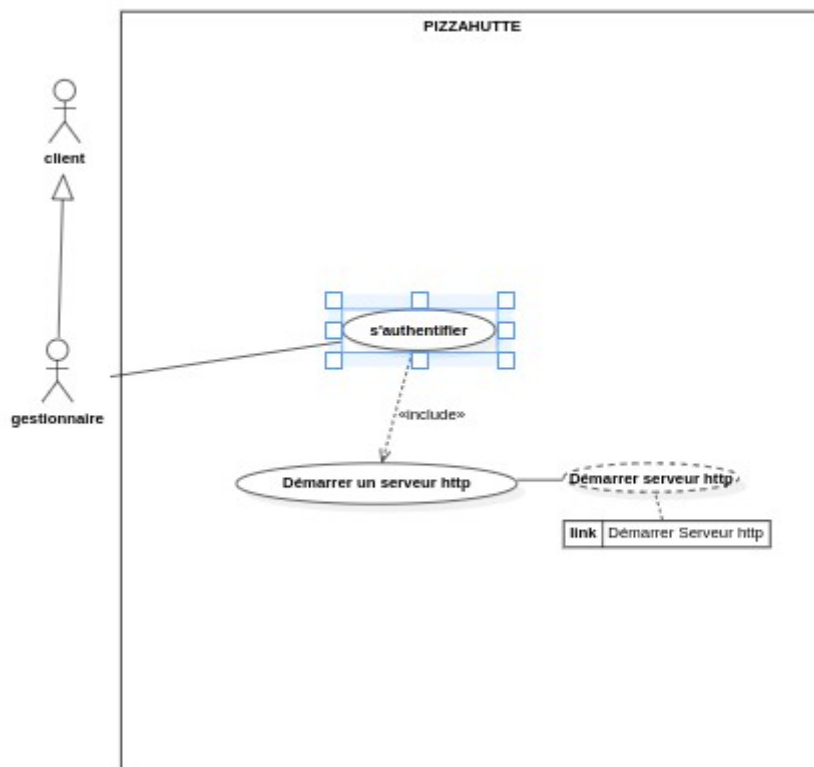
Sous la direction de Rossi Oddet

Table des matières

1. ANALYSE ET CONCEPTION.....	3
1.1. Diagramme des cas d'utilisation.....	4
2. S'authentifier.....	5
2.1. Premier jet.....	5
2.1.1. Diagramme de séquence.....	5
2.1.2. Diagramme des classes participantes.....	5
2.2. Second jet.....	7
2.2.1. Diagramme de séquence corrigé et approfondi.....	7
2.2.2. Diagramme des classes participantes.....	9
3. Choix des outils technologiques.....	10
4. Organisation et planification du projet.....	11
5. DEVELOPPEMENT.....	12
5.1. Mise en place de l'environnement de travail.....	13
5.1.1. Structuration de l'arborescence du projet.....	13
5.1.2. Création du dépôt github et clonage du projet vierge.....	13
5.1.3. Création du projet node via npm.....	13
5.1.3.1. installation des dépendances.....	13
5.1.3.1.1. Paramétrage du fichier de configuration package.json.....	13
Affectation de nodemon à la commande start de l'objet script.....	13
5.1.3.1.2. Prise en charge des modules ECMAScript (import, export).....	13
5.1.3.2. Consignation de la procédure dans un script BASH. Permettra de générer immédiatement le projet en cas de « pépin ».....	13
6. DEVELOPEMENT DE L'APPLICATION PIZZAHUTTE.....	15
6.1. Pf1 : Démarrer un serveur.....	16
6.1.1. Script de contrôle principal Wwww.js.....	16
6.1.2. Interface de gestion de l'application Express app.js.....	17
6.1.3. Script de routage de la page d'accueil authentification.js.....	18
6.1.4. Script de routage de la page d'accueil client users.js.....	18
7. annexes.....	19
7.1.1. Vues à générer par le moteur de vue Pug.....	19
7.1.1.1. Layout.pug.....	19
7.1.1.2. Authentification .pug.....	19
7.1.1.3. Error.pug.....	19

1. ANALYSE ET CONCEPTION

1.1. Diagramme des cas d'utilisation



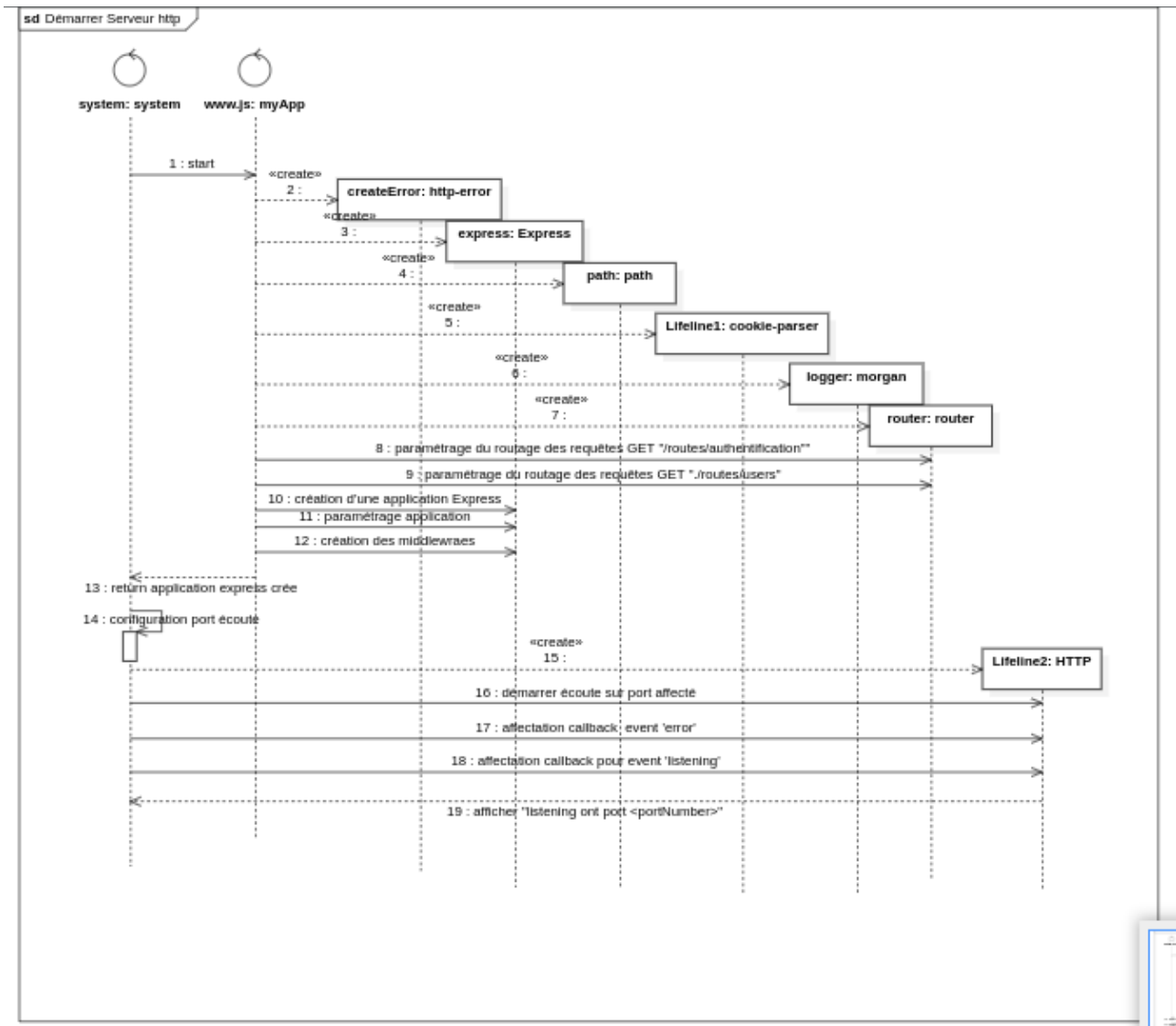
PIZZAHUTTE

Création d'une application Javascript FullStack complète

2. S'authentifier

2.1. Premier jet

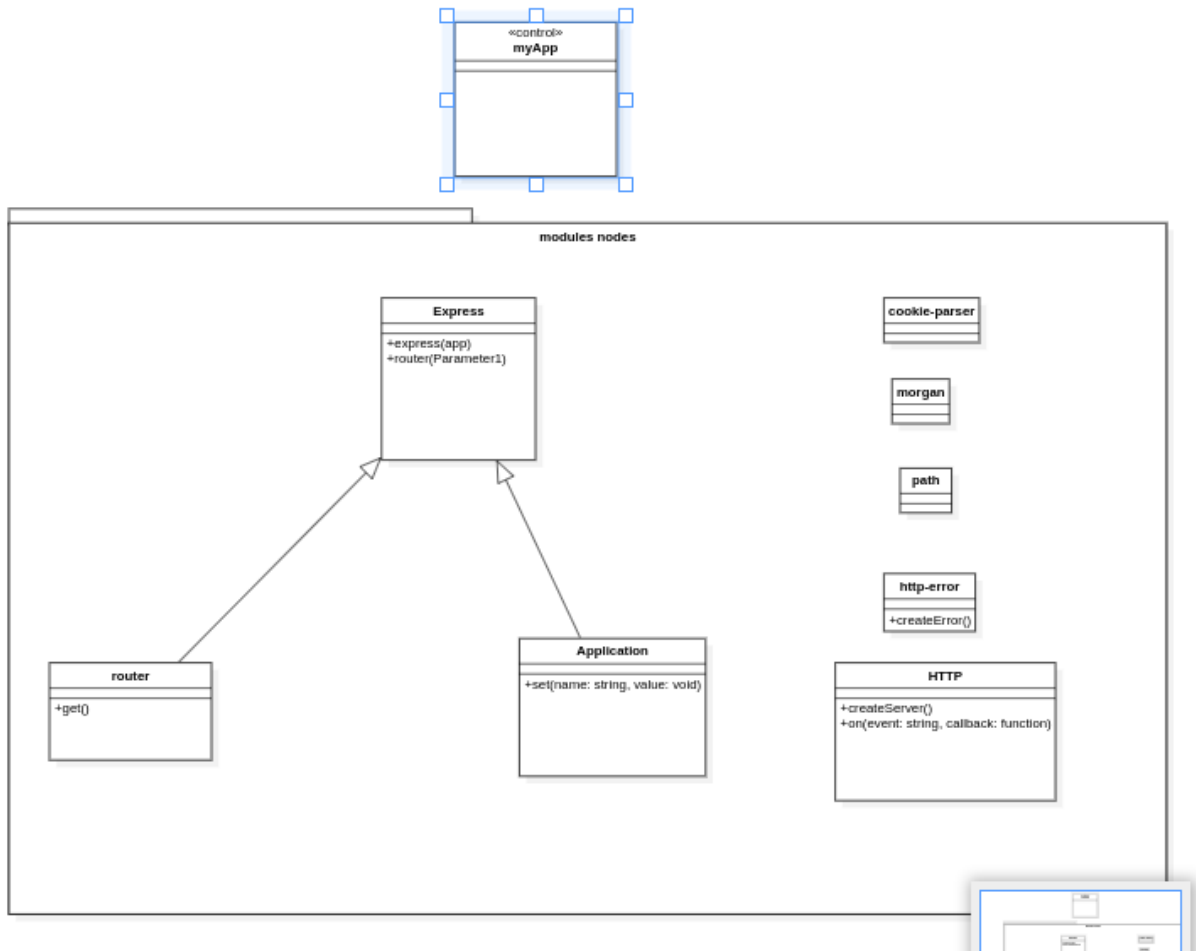
2.1.1. Diagramme de séquence



2.1.2. Diagramme des classes participantes

PIZZAHUTTE

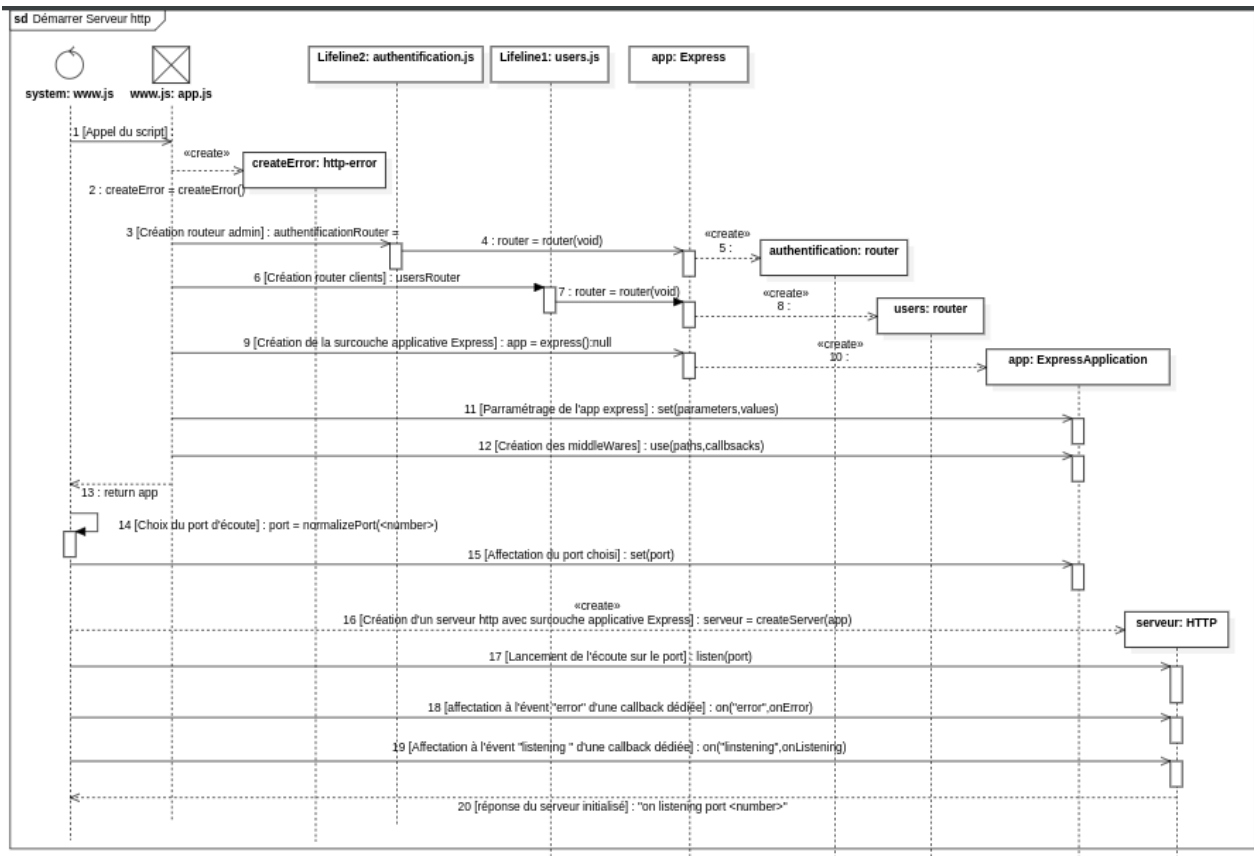
Création d'une application Javascript FullStack complète

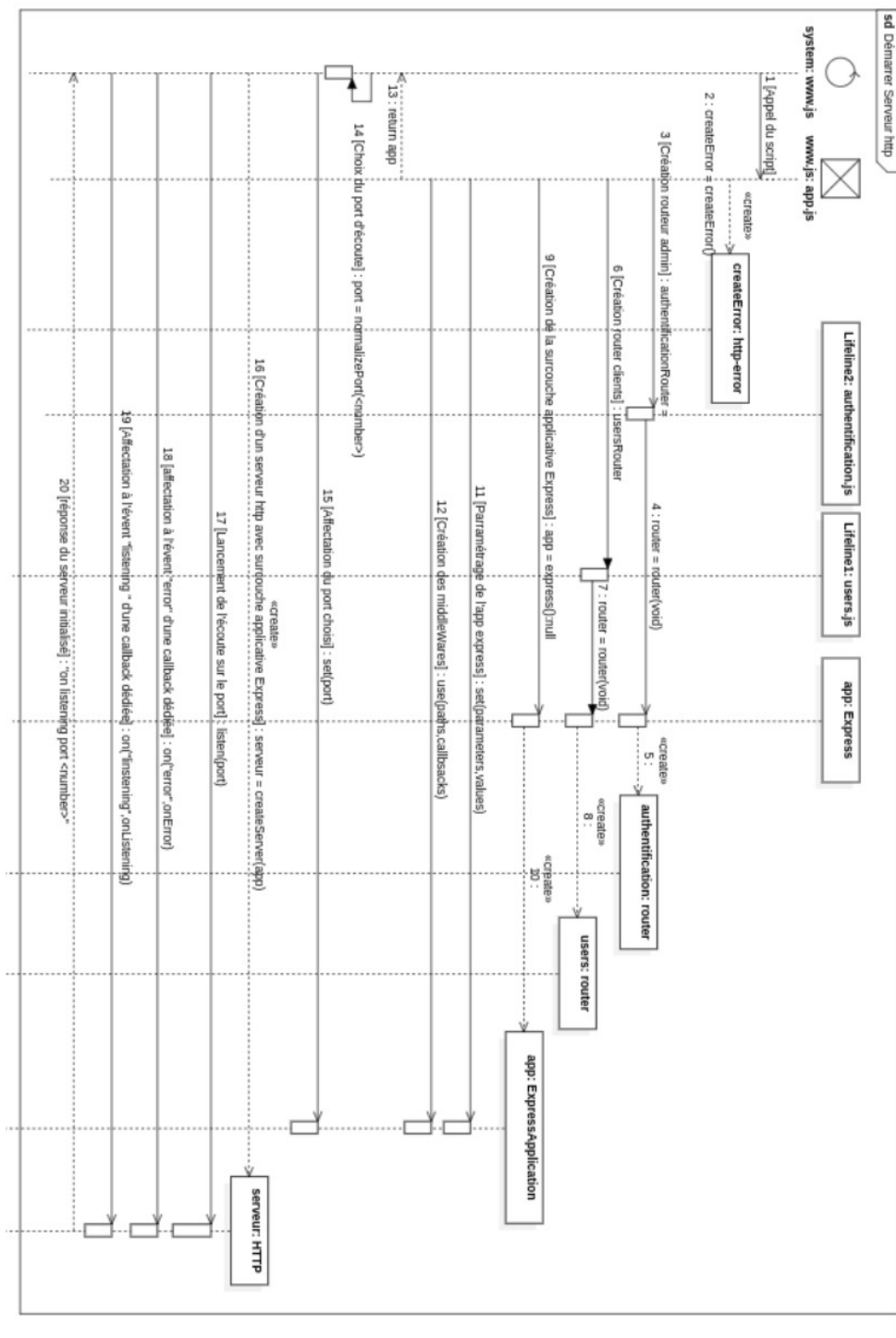


2.2. Second jet

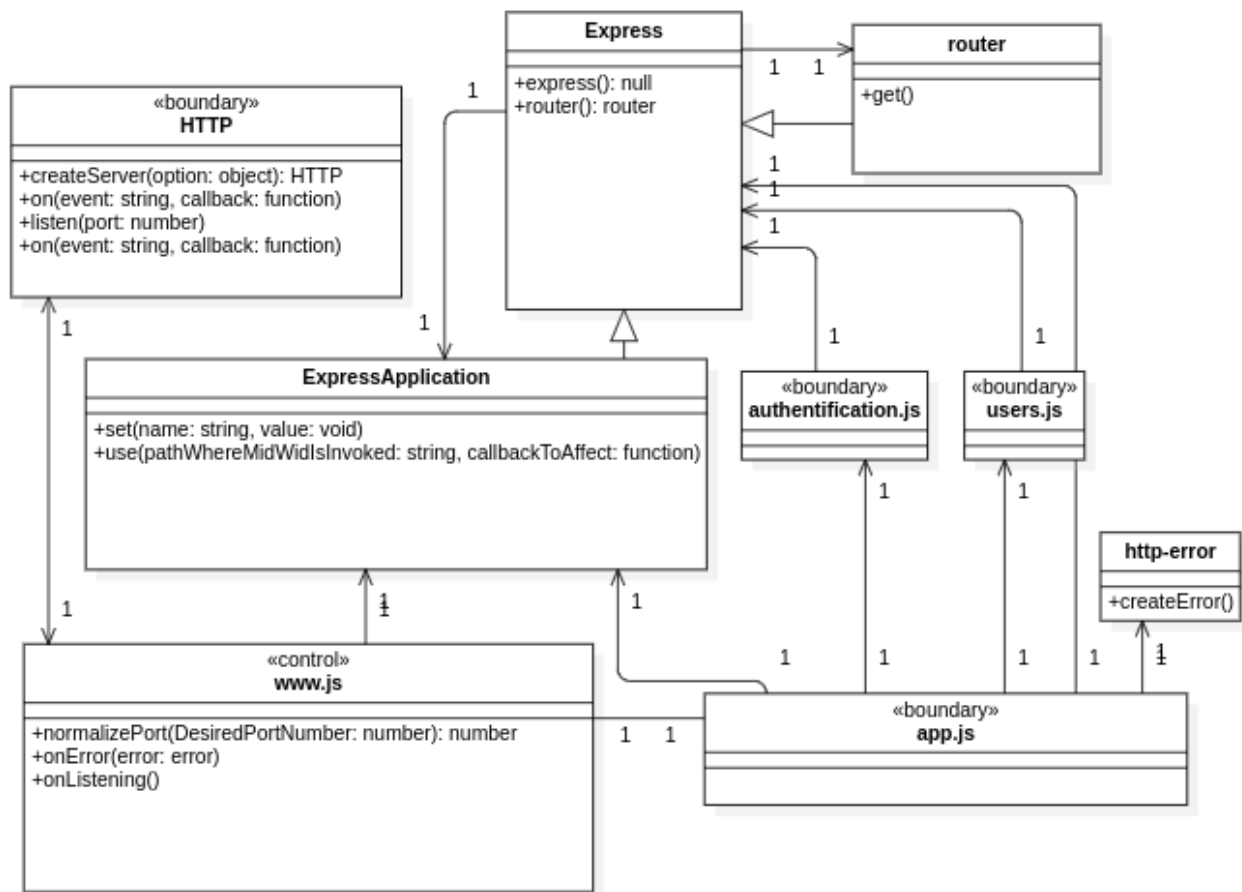
Nous allons affecter les méthodes des classes aux différentes interactions entre les objets de l'application

2.2.1. Diagramme de séquence corrigé et approfondi





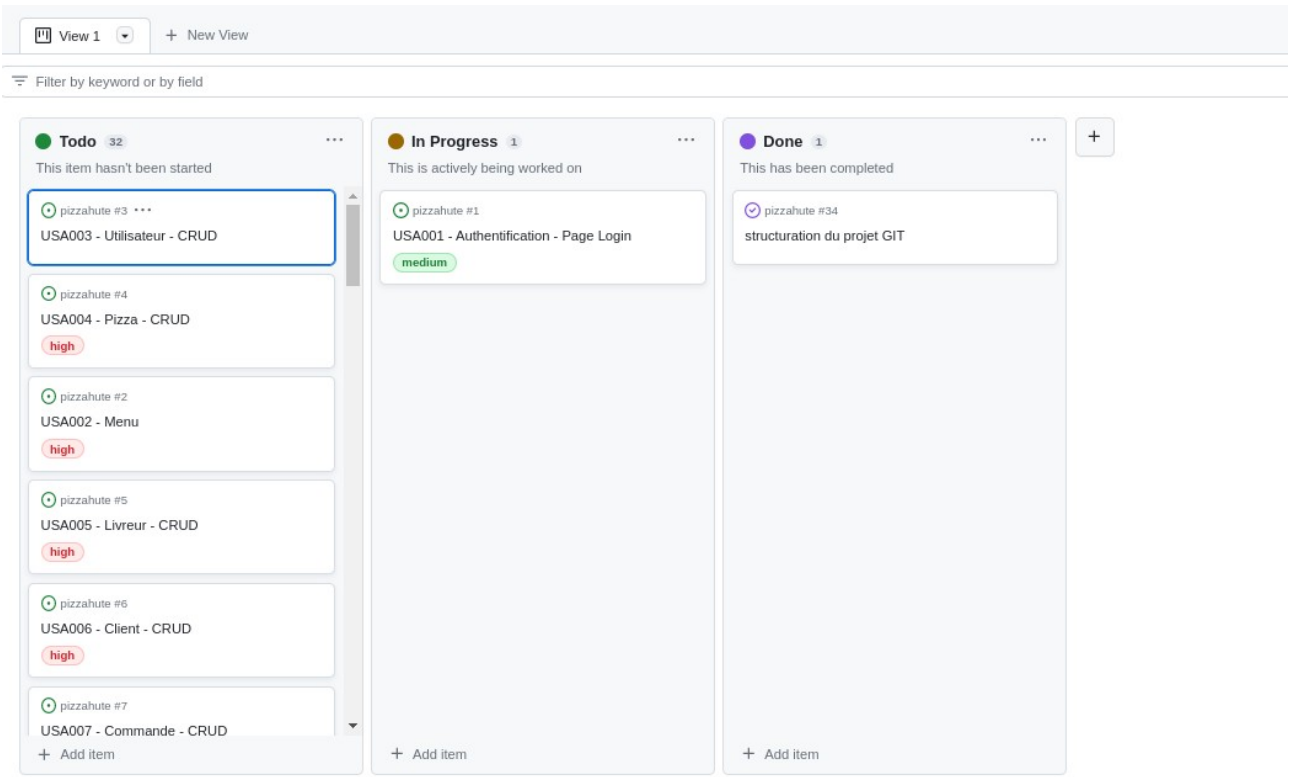
2.2.2. Diagramme des classes participantes



3. Choix des outils technologiques

4. Organisation et planification du projet

4.1. Découpage des tâches sous la forme KANBAN



5. DEVELOPPEMENT


```

echo "[Création d'un .gitignore"

express --pug -css sass --view pug --git $uservar
cd $uservar

echo "[Installation des dépendances..."
echo "[      --nodemon"
npm install nodemon --save-dev
echo "[      --bootstrap"
npm install bootstrap --save
echo "[      --socket.io"
npm install socket.io --save
echo "[      --body-parser"
npm install body-parser --save

echo "[Test du serveur"
echo -e "\033[31m [Serveur en attente : veuillez entrer dans un navigateur 'localhost:3000'\033[1;32m"
DEBUG=$uservar:* npm start

```

6. DEVELOPEMENT DE L'APPLICATION PIZZAHUTTE

6.1. Pf1 : Démarrer un serveur

6.1.1. Script de contrôle principal Www.js

```
#!/usr/bin/env node

/**
 * Module dependencies.
 */

var app = require("../app");
var debug = require("debug")("pizzahutte:server");
var http = require("http");

/**
 * Get port from environment and store in Express.
 */

var port = normalizePort(process.env.PORT || "2000");
app.set("port", port);

/**
 * Create HTTP server.
 */

var server = http.createServer(app);

/**
 * Listen on provided port, on all network interfaces.
 */

server.listen(port);
server.on("error", onError);
server.on("listening", onListening);

/**
 * Normalize a port into a number, string, or false.
 */

function normalizePort(val) {
  var port = parseInt(val, 10);

  if (isNaN(port)) {
    // named pipe
    return val;
  }

  if (port >= 0) {
    // port number
    return port;
  }

  return false;
}

/**
 * Event listener for HTTP server "error" event.
 */

function onError(error) {
  if (error.syscall !== "listen") {
    throw error;
  }

  var bind = typeof port === "string" ? "Pipe " + port : "Port " + port;

  // handle specific listen errors with friendly messages
  switch (error.code) {
    case "EACCES":
      console.error(bind + " requires elevated privileges");
      process.exit(1);
      break;
  }
}
```

PIZZAHUTTE

Création d'une application Javascript FullStack complète

```

        case "EADDRINUSE":
            console.error(bind + " is already in use");
            process.exit(1);
            break;
        default:
            throw error;
    }
}

/**
 * Event listener for HTTP server "listening" event.
 */

function onListening() {
    var addr = server.address();
    var bind = typeof addr === "string" ? "pipe " + addr : "port " + addr.port;
    debug("Listening on " + bind);
}

```

6.1.2.Interface de gestion de l'application Express app.js

```

var createError = require("http-errors");
var express = require("express");
var path = require("path");
var cookieParser = require("cookie-parser");
var logger = require("morgan");

var authenticationRouter = require("./routes/authentication.js");
var usersRouter = require("./routes/users");
const exp = require("constants");

var app = express();

// view engine setup
app.set("views", path.join(__dirname, "views"));
app.set("view engine", "pug");

app.use(logger("dev"));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(
    "/css",
    express.static(
        path.join(__dirname, "node_modules", "bootstrap", "dist", "css")
    )
);
app.use(express.static(path.join(__dirname, "public")));
app.use(
    express.static(
        path.join(__dirname, "node_modules", "bootstrap", "dist", "css")
    )
);
/*routes*/
app.use("/", authenticationRouter);
app.use("/users", usersRouter);

// catch 404 and forward to error handler
app.use(function (req, res, next) {
    next(createError(404));
});

```

PIZZAHUTTE

Création d'une application Javascript FullStack complète

```
// error handler
app.use(function (err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get("env") === "development" ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render("error");
});

module.exports = app;
```

6.1.3. Script de routage de la page d'accueil authentication.js

```
var express = require("express");

var router = express.Router();

/* GET home page. */
router.get("/", function (req, res, next) {
  res.render("authentication", { title: "Pizzahutte" });
});

module.exports = router;
```

6.1.4. Script de routage de la page d'accueil client users.js

```
var express = require('express');
var router = express.Router();

/* GET users listing. */
router.get('/', function(req, res, next) {
  res.send('respond with a resource');
});

module.exports = router;
```

7. annexes

7.1.1. Vues à générer par le moteur de vue Pug

7.1.1.1. *Layout.pug*

```
doctype html
html
  head
    title= title
    link(rel='stylesheet', href='/stylesheets/style.css')
    link(rel='stylesheet', href='/bootstrap.min.css')
    meta(name="viewport" content="width=device-width, initial-scale=1.0")
  body
    block content
```

7.1.1.2. *Authentication .pug*

```
extends layout

block content
  header
    div(class="mx-auto mb-5")
      img(src="images/logo-no-background-small.png" class="img-fluid" alt="")
      h1(class="h1 text-center text-secondary") PIZZAHUTTE
  main
    div(class="container text-center")
      form(class="m-auto")
        // Email input
        .form-outline.mb-4
          input#form2Example1.form-control(type='email')
          label.form-label(for='form2Example1') Email
        // Password input
        .form-outline.mb-4
          input#form2Example2.form-control(type='password')
          label.form-label(for='form2Example2') Mot de passe
        // 2 column grid layout for inline styling
        div.d-flex.flex-row
          button.btn.btn-primary.btn-block.mp-10.text-center(type='button') Se
connecter
      .text-center.mt-3
        a.text-center(href='#!') Mot de passe oublié ?
```

7.1.1.3. *Error.pug*

```
extends layout

block content
  h1= message
  h2= error.status
  pre #{error.stack}
```

PIZZAHUTTE

Création d'une application Javascript FullStack complète

