Instituto Tecnológico de Costa Rica

Ingeniería en Computadores

Bases de Datos – CE3101

Profesor: Marco Rivera Meneses

Tarea Corta #1: TECBOx

Realizado por:

Alex Marín López	2015040318
Fabian Montero Villalobos	2016121558
Mariano Muñoz Masís	2016121607
Melany Acuña Vega	2016142297

I Semestre 2020

CONTENTS

DESCRIPCIÓN DE LOS MÉTODOS IMPLEMENTADOS.

ASP.NET

ASP.NET se utilizó para la construcción del API. Para cada endpoint del API hay un controlador de ASP.NET escrito en C#, que contiene funciones para manejar solicitudes HTTP de diferentes clientes. A continuación, un ejemplo básico de un método para manejar una solicitud DELETE:

```
// DELETE: api/Workers/id
[Route("api/Workers/{id:int}")]
[HttpDelete]
public HttpResponseMessage Delete(int id)
{
      db.workers.delete(id);
      HttpResponseMessage response =
         Request.CreateResponse(HttpStatusCode.Created);
      response.Headers.Add("Access-Control-Allow-Origin", "*");
      response.Content = new StringContent("{\"error\": null}");
      return response;
}
```

En este caso, la función recibe el ID de un trabajador y lo elimina de la base de datos, luego de eso, construye una respuesta y coloca los headers necesarios para que pueda ser leída fácil y finalmente la envía.

TYPESCRIPT:

- **Métodos GET y POST para los componentes de la página.**

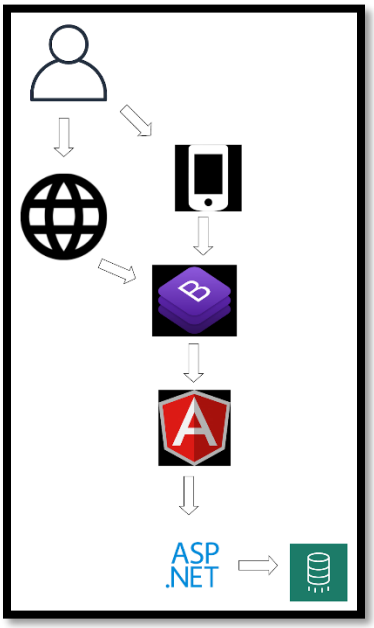
La estructura de los métodos desarrollados para realizar peticiones al API es un estándar que varía únicamente la extensión de la dirección a la que se realiza el “request” y algunos parámetros secundarios, como es el caso de POST que requiere un **Data Payload** o paquete de datos como parámetro. Por ejemplo, la totalidad de métodos GET comparten una misma forma, sin embargo, se toma en consideración lo que se necesita mostrar en pantalla para solicitar información. De forma similar los POST utilizan información de interés recopilada de inputs o entradas en la página y lo envuelven para ser enviado y almacenado o procesado.

DESCRIPCIÓN DE LAS ESTRUCTURAS/ENTIDADES DESARROLLADAS.

JSON

A nivel del API se desarrolló una estructura de Json para la comunicación API-Página web. La cual define el estándar de mensajes requerido para una correcta recolección y almacenamiento de información.
En términos generales la estructura de Json comienza con una token de error, el cual indica si la petición fue tramitada de forma correcta o incorrecta, sin embargo a partir de este punto difiere para el tipo de petición que se realiza.
Json para GET requests:
Seguido del token de error, los Json que retorna el API tras una solicitud de tipo GET, contienen un identificador que denota de donde fue tomada la información (Trabajadores, paquetes, compras, vendedores, etc) seguido de la información solicitada.
Jason para POST requests:
Las estructuras de Json para un post request en el API únicamente se encargan de devolver un mensaje de error, en caso de presentarse alguno o bien un error ninguno en caso de que no.

DESCRIPCIÓN DETALLADA DE LA ARQUITECTURA DESARROLLADA.



La arquitectura del sistema consiste en una capa de visualización dinámica con Bootstrap, una capa de lógica con Angular.JS y un servicio REST con ASP.NET.

La capa de visualización es desplegada en un navegador web o bien en una app para uso del bodeguero.

Con esta arquitectura, el usuario interactúa con la base de datos por medio de la app o su explorador, con la ventaja de que hay un solo diseño para ambas vistas.

La implementación de Angular mediante su framework nos permite desarrollar aplicaciones web de una sola página, su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC). Angular CLI nos brinda un conjunto de bibliotecas y comandos para generar proyectos ya integrados en sí mismo.

Bootstrap nos permite utilizar estilos y templates pre-hechos, para estilizar los proyectos, Bootstrap complementa al framework de Angular en su faltante de diseño, además de la única implementación en el proyecto lo que nos permite eficientizar el uso de recursos.

PROBLEMAS SIN SOLUCIÓN

PROBLEMAS ENCONTRADOS

Algunos de los problemas que se presentaron al desarrollar el modelo de comunicación del HttpClient de angular con el API son el uso de CORS headers y errores generados por Cross origin requests. Ambas situaciones se presentan cuando se realiza una petición (POST, GET, entre otras) de un protocolo a otro (http a https o viceversa e inclusive a alguno que no se menciona aquí).
Inicialmente como método de solución temporal se realizan las peticiones a un sitio de “mock up” (https://jsonplaceholder.typicode.com) que acepta peticiones de sitios con protocolo http. Sin embargo esta solución temporal no es suficiente para el desarrollo completo, al momento de conectar la página con su correspondiente API se presenta nuevamente el problema, por lo que se procede a buscar una solución más permanente.

En el desarrollo de la aplicación para el mensajero aparecen nuevas dificultades. Fallas en la sincronización con el “Gradle” fue posiblemente la más complicada de resolver. Problemas que involucran el gradle dentro del ambiente de desarrollo de Android Studio se pueden deber a, pero no limitado, errores en la construcción, fallas en la sincronización o bien por archivos en el cache corruptos. El primer intento de solución que se da es borrar el folder ***./gradle*** del proyecto, lo cual no funciona. Como un segundo intento se elimina el folder de ***cache*** en la carpeta ***./gradle*** en el directorio de usuario. Esto tampoco soluciona el problema por lo que se procede a elimina el folder completo de ***./gradle*** del directorio de usuario (cabe destacar que realizar esta acción es costosa pues gradle suele ser un archivo pesado y puede tomar tiempo en descargar todas las dependencias y elementos requeridos). Por último, se borra la carpeta de proyecto y se crea uno nuevo, es mediante esto que se logra una sincronización y construcción exitosa del gradle.

Al implementarse un **“WebView”** para la aplicación se encuentra el error : net::ERR_CLEARTEXT_NOT_PERMITTED.
Posibles soluciones para este problema se encuentran en StackOvewflow, donde sugieren 2 acciones.

1. Agregar android:usesCleartextTraffic="true" en el application tag del archivo androidManifest.
2. En caso de tener la configuración de red de la siguiente forma android:networkSecurityConfig="@xml/network_security_config" basta con colocar las siguientes lineas de codigo en el activitymain.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        ....
        ....
    </domain-config>

    <base-config cleartextTrafficPermitted="false"/>
</network-security-config>
```

- **Errores en las animaciones de Bootstrap:**

Al importar algun template de Bootstrap, ocurría que las animaciones respectivas al template no se ejecutaban, aun cuando los archivos de bootstrap.min.css, jquery.slim.js y bootstrap.bundle.min.js eran correctamente llamados.
El error se soluciono al importar de manera global estos archivos utilizando los comandos de consola *npm install --save jquery*, dentro de la carpeta del proyecto, junto con los módulos de desarrollo de bootstrap, al reiniciar el proyecto los módulos aparecen dentro de el angular.json y la carpeta “node_modules”.

DOCUMENTACIÓN DE EVIDENCIA DEL TRABAJO EN EQUIPO.

ACTIVIDADES PLANEADAS Y SU RESPONSABLE. (PLAN DE TRABAJO)

Inicialmente se crea un plan de trabajo y se asignan tareas a cada integrante del grupo, lo cual se ve reflejado en el sistema de administración de proyectos “Trello” (https://trello.com/b/TXWwX1y7/tarea-1).
Como parte de las etapas de desarrollo se crean historias de usuario para modelar el contenido del software. Estas se presentan a continuación:

Clientes/Usuarios

1. Yo como usuario quiero poder ver productos disponibles.
2. Yo como usuario quiero ser capaz de rastrear mis paquetes.
3. Yo como usuario quiero tener la facilidad de buscar productos en la página.
4. Yo como usuario quiero poder tener un casillero.

Administrador

1. Yo como administrador quiero poder manejar los roles dentro de la página.
2. Yo como administrador quiero gestionar a mis empleados.
3. Yo como administrador quiero gestionar paquetes.
4. Yo como administrador quiero gestionar rutas.

Mensajero

1. Yo como mensajero quiero poder acceder desde mi teléfono celular.
2. Yo como mensajero quiero poder confirmar el estado de los paquetes.

EVIDENCIA DE USO DE UN MANEJADOR DE CÓDIGO (SE RECOMIENDA GITHUB).

En esta sección se presenta el link al repositorio en GitHub como evidencia de uso de manejador de código.

https://github.com/MarianoIDC/Tarea-Corta-master

CONCLUSIONES Y RECOMENDACIONES DEL PROYECTO.

El uso de Frameworks facilita la creación de recursos web, debido a la completitud que estos presentan, además que al estandarizarse el uso de estos el soporte que se le pude dar aumenta, debido a la cantidad de usuarios que se presentan, además que la plataforma sea de código abierto como lo es en este caso Angular optimiza el encuentro de fallos y sus respectivas soluciones.

Además, debemos resaltar el uso de Bootstrap en este proyecto, al generar plantillas el enfoque a la pagina web se vuelve mas sencillo, debido a que el curso en si es de bases de datos y no de paginas web. La implementación de plantillas junto con la integración de Angular facilita el desarrollo de paginas interactivas e interesantes para el usuario.

BIBLIOGRAFÍA CONSULTADA EN TODO EL PROYECTO.

Documentación Angular.cli
Recuperado de: https://angular.io/

Bootstrap Themes & Templates
Recuperado de: https://getbootstrap.com/docs/4.1/getting-started/introduction/

HTML y CSS curso práctico
Recuperado de: https://www.youtube.com/watch?v=rr2H086z16s

WebView showing ERR_CLEARTEXT_NOT_PERMITTED although site is HTTPS [duplicate]
Recuperado de: https://stackoverflow.com/questions/52707918/webview-showing-err-cleartext-not-permitted-although-site-is-https

WebView and localhost
Recuperado de: https://stackoverflow.com/questions/4336394/webview-and-localhost

Accessing localhost:port from Android emulator
Recuperado de: https://stackoverflow.com/questions/6760585/accessing-localhostport-from-android-emulator

Dev Tip: How to view localhost web apps on your phone
Recuperado de: https://medium.com/@prowe214/tip-how-to-view-localhost-web-apps-on-your-phone-ad6b2c883a7c

Error running MyActivity: Gradle project sync failed. Please fix your project and try again
Recuperado de: https://stackoverflow.com/questions/24778598/error-running-myactivity-gradle-project-sync-failed-please-fix-your-project-an/33713167

Recuperado de: https://stackoverflow.com/questions/29808199/error-running-android-gradle-project-sync-failed-please-fix-your-project-and-t

Error:Unexpected lock protocol found in lock file. Expected 3, found 0.
Recuperado de: https://stackoverflow.com/questions/43010547/errorunexpected-lock-protocol-found-in-lock-file-expected-3-found-0?rq=1
WebView
Recuperado de: https://codinginflow.com/tutorials/android/webview

Error: "error:cannot resolve symbol 'WebViewClient'
Recuperado de: https://stackoverflow.com/questions/43857126/error-errorcannot-resolve-symbol-webviewclient/43857510

Error:Unexpected lock protocol found in lock file. Expected 3, found 0
Recuperado de: https://stackoverflow.com/questions/31743942/errorunexpected-lock-protocol-found-in-lock-file-expected-3-found-0

Set CORS header in ASP.NET HTTP response
Recuperado de: https://stackoverflow.com/questions/61928644/set-cors-header-in-asp-net-http-response/

Android WebView dont work for localhost
Recuperado de: https://www.codeproject.com/Questions/876885/Android-WebView-dont-work-for-localhost