UNIVERSIDAD DE BURGOS

Escuela Politécnica Superior

Gº en Ingeniería Informática

****

**TFG Ingeniería Informática:**

**GII17.0T Trayectorias Semánticas**

Presentado por Hector Cogollos Adrian

en Burgos el 2 julio de 2019

Tutores D. Bruno Baruque Zanón

y D. Santiago Porras Alfonso

# Índice General

[Índice General 1](#_Toc12813256)

[Índice de figuras 3](#_Toc12813257)

[Plan de Proyecto Software 4](#_Toc12813258)

[A.1. Introducción 4](#_Toc12813259)

[A.2. Planificación temporal 4](#_Toc12813260)

[A.3. Estudio de viabilidad 4](#_Toc12813261)

[Viabilidad económica 4](#_Toc12813262)

[Viabilidad legal 4](#_Toc12813263)

[Especificación de Requisitos 5](#_Toc12813264)

[B.1. Introducción 5](#_Toc12813265)

[B.2. Objetivos generales 5](#_Toc12813266)

[B.3. Catálogo de requisitos 5](#_Toc12813267)

[B.4. Especificación de requisitos 5](#_Toc12813268)

[Especificación de diseño 6](#_Toc12813269)

[C.1. Introducción 6](#_Toc12813270)

[C.2. Diseño de datos 6](#_Toc12813271)

[C.3. Diseño procedimental 6](#_Toc12813272)

[C.4. Diseño arquitectónico 6](#_Toc12813273)

[Documentación técnica de programación 7](#_Toc12813274)

[D.1.Introducción 7](#_Toc12813275)

[D.2. Estructura de directorios 7](#_Toc12813276)

[D.3. Manual del programador 7](#_Toc12813277)

[D.4. Compilación, instalación y ejecución del proyecto 7](#_Toc12813278)

[D.5. Pruebas del sistema 7](#_Toc12813279)

[Documentación de usuario 8](#_Toc12813280)

[E.1. Introducción 8](#_Toc12813281)

[E.2. Requisitos de usuarios 8](#_Toc12813282)

[E.3. Instalación 8](#_Toc12813283)

[E.4. Manual del usuario 8](#_Toc12813284)

# Índice de figuras

# Plan de Proyecto Software

## A.1. Introducción

En este apartado se expone la planificación temporal del proyecto y el estudio de viabilidad económico y legal. En el apartado de planificación temporal se explica que técnicas se han utilizado y como se han utilizado. En viabilidad económica veremos los costes de la puesta en marcha del proyecto y en viabilidad legal estudiaremos si las licencias de todos los componentes del proyecto son compatibles legalmente.

## A.2. Planificación temporal

La planificación temporal está basada en **Scrum** una metodología ágil para la gestión de proyectos. Este ha sido adaptado a las circunstancias del proyecto ya que la disponibilidad horaria no ha sido la misma a lo largo del desarrollo y a el hecho de que nos encontramos en un contexto educativo. En este contexto no hay un cliente y tampoco hay reuniones diarias ya que solo hay un desarrollador.

Algunas de las cosas que hay que tener en cuenta antes de pasar a ver los detalles de la planificación son las siguientes:

* El proyecto se divide en sprints o hitos, cada hito tiene una duración similar, aunque adecuándose a las circunstancias.
* Los sprints tienen tareas asociadas con un cálculo de duración en fracciones de tiempo indeterminadas.
* Se persigue que cada sprint tenga una carga de trabajo similar a la del resto de sprints.
* No se ha podido determinar el grado de cumplimiento de los tiempos debido a la falta de horarios.

### Sprint 1

Su finalidad es investigar acerca del proyecto aspectos tales como artículos sobre el tema de las trayectorias semánticas, aplicaciones relacionadas, herramientas para su desarrollo, etc.

Fecha de fin: 30/11/2018

|  |  |
| --- | --- |
| Tarea | Estimación |
| Filtrar trayectorias por tiempo | 5 |
| Extraer datos de las rutas | 3 |
| Estudiar proyectos similares | 3 |
| Estudiar artículos de trayectorias | 13 |
| Investigar herramientas | 8 |
| Añadir memoria al repositorio | 1 |

## Sprint 2

En este sprint se buscaba prepara algunos de los elementos básicos para la aplicación. Fundamentalmente los sistemas de bases de datos y las bases de datos geoespaciales.

Fecha de fin: 17/12/2018

|  |  |
| --- | --- |
| Tarea | Estimación |
| Documentarse sobre la instalación de PostGIS | 3 |
| Instalar PostGIS | 3 |
| Estudiar como cagar datos en PostGIS | 5 |
| Buscar información de como cargar los datos de OSM | 5 |
| Cargar datos de OSM | 3 |

## Sprint 3

Investigar los datos de OSM para ver cuáles son útiles para nuestro proyecto y tratar de construir trayectorias

Fecha de fin: 11/01/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Seleccionar datos relevantes de OSM | 8 |
| Anotar Trayectorias | 5 |

Sprint 4

Procesar las rutas para ver que tenemos y si se pueden relacionar con los datos de OSM.

Fecha de fin: 01/02/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Preprocesar las rutas | 13 |

## Sprint 5

Implementar un algoritmo de detección de paradas.

Fecha de fin: 15/02/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Obtener stops en las rutas | 13 |

## Sprint 6

Implementar un segundo algoritmo de detección de paradas.

Fecha de fin: 01/03/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| implementación de un segundo algoritmo de detección de paradas | 5 |

## Sprint 7

El objetivo de este sprint es crear un primer modelo para los datos de la aplicación que son las trayectorias conceptuales y crear una primera base de datos que las contenga.

Fecha de fin: 14/03/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Refactorizar código | 5 |
| Crear Modelo Conceptual | 5 |
| Crear Base de Datos | 8 |

### Sprint 8

Modificar el diseño de la base de datos del anterior sprint, empezar con la documentación e implementar funciones para cargar datos en la base de datos.

Fecha de fin: 29/03/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Hacer documentación | 5 |
| Adaptar la base de datos a los cambios | 2 |
| Implementar la carga de datos en la base de datos | 8 |

### Sprint 9

Cargar los datos en la base de datos y ampliar la documentación.

Fecha de fin: 10/04/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Memoria rellenar conceptos teóricos | 5 |
| Cargar datos de la BD | 3 |

### Sprint 10

Preparar un servidor con la aplicación de Nominatim para la busqueda de puntos de interés y estudiar un algoritmo de predicción.

Fecha de fin: 26/04/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Manual de Instalación de Nominatim | 5 |
| Instalar Nominatim | 8 |
| Lectura de datos de Nominatim | 5 |
| Estudiar el algoritmo de predicción | 3 |

### Sprint 11

Probar algoritmos de *clustering* y el clasificador estudiado en el sprint anterior.

Fecha de fin: 17/05/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Probar Clustering | 8 |
| Probar Clasificador de árbol | 8 |

### Sprint 12

Resolver problemas que se podían apreciar en el clasificador dado que había demasiadas clases para predecir y muy pocos datos.

Fecha de fin: 30/05/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Crear arboles tipo | 5 |
| Agrupan clases por grupos | 13 |
| Resolver el problema de los edificios sin tipo | 13 |

### Sprint 13

Aprender como realizar aplicaciones web mas concretamente con el *framework* de Flask y crear un diseño base de la aplicación.

Fecha de fin: 03/06/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Diseño Base de la aplicación web | **5** |
| Aprender Flask | 5 |

### Sprint 14

Avanzar a la memoria y los anexos y tratar de emplear un algoritmo de *clustering* utilizando matrices de adyacencia.

Fecha de fin: 11/06/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Anexo B: B-1, B-2, B-3 | **8** |
| Anexo C: Diagramas de casos de uso | 5 |
| Crear un algoritmo de clustering | **8** |
| Memoria: Conceptos teóricos | 5 |
| Memoria: Objetivos | **3** |
| Memoria: Introducción | 3 |

### Sprint 15

Finalizar la memoria la pagina web y los anexos.

Fecha de fin: 02/07/2019

|  |  |
| --- | --- |
| Tarea | Estimación |
| Terminar la memoria | **8** |
| Terminar página web | 13 |

En la siguiente tabla tenemos las sumas de los tiempos de cada sprint y el total, se si sacamos la media de los tiempos nos da que cada sprint debería durar 18,2 fracciones de tiempo.

|  |  |
| --- | --- |
| Sprint 1 | 38 |
| Sprint 2 | 19 |
| Sprint 3 | 13 |
| Sprint 4 | 13 |
| Sprint 5 | 13 |
| Sprint 6 | 5 |
| Sprint 7 | 18 |
| Sprint 8 | 15 |
| Sprint 9 | 8 |
| Sprint 10 | 21 |
| Sprint 11 | 16 |
| Sprint 12 | 31 |
| Sprint 13 | 10 |
| Sprint 14 | 32 |
| Sprint 15 | 21 |
| Total | 273 |

## A.3. Estudio de viabilidad

### Viabilidad económica

En este apartado de viabilidad económica se va a a calcular los costes que tendría el desarrollo de esta aplicación para una empresa. Y los costes de explotación anuales de la aplicación plenamente operativa y con capacidad para trabajar con datos de todo el planeta. Los requisitos del sistema están basados en recomendaciones de Nominatim para manejar una base de datos de OSM completa.

#### Costes de desarrollo

Primero se calculan los costes que supone contratar a un desarrollador, para calcular esto se estima que un programador cobra 1500€ brutos y se estima el coste de la seguridad social en un 33%. También se tiene en cuenta que la duración del proyecto son 30 horas por crédito siendo 12 créditos. Esto hacen 360 horas que son aproximadamente dos meses a jornada completa.

|  |  |
| --- | --- |
| Sueldo bruto | 1.500 € |
| Seguridad social | **495€** |
| Total | **3.990 €** |

A esto hay que añadir los costes derivados del equipo necesario para el desarrollo del software. Se calcula un plazo de amortización de 5 años para los equipos informáticos. Partimos de que un portátil promedio para este fin tiene un coste de 800€ con sistema operativo incluido.

|  |  |
| --- | --- |
| Office 365 Empresa Premium | 21,00€ |
| Visual Studio Code | **0,00€** |
| Ordenador | **26,66€** |
| Total | **47,66€** |

Esto hace un coste total de desarrollo de **4.037,66€**

#### Costes de explotación

Para esto vamos a calcular el coste mensual que tiene un servidor Con 32GB de memoria RAM, 8 vCPU y una capacidad de 2TB de disco duro sólido. Los costes son mensuales, están estimados para la plataforma Azure de Microsoft y pueden variar sensiblemente.

|  |  |
| --- | --- |
| Servidor virtual | 240,93€ |
| 2 TB disco duro SSD | **198,62€** |
| Ubuntu server 18.04 | **0€** |
| Total | **439,55€** |

El coste anual de explotación se sitúa en **5.274,60€**

### Viabilidad legal

En este apartado se estudia la viabilidad legal de la aplicación. Para esto se tiene en cuenta en primer lugar las licencias del software integrado en nuestra aplicación.

|  |  |
| --- | --- |
| Pandas | BSD |
| GeoPandas | **BSD 3** |
| SQLAlchemy | **MIT** |
| Flask | **BSD** |
| Prefixspan | **MIT** |
| BootStrap | **MIT** |

Todas ellas pueden usarse en una aplicación con licencia AGPL3 por lo que no supone ningún problema. Esta licencia es similar a GPL3, pero está pensada para aplicaciones Web.

El otro punto problemático puede ser el tratamiento de datos, pero al no contener datos personales ni que permitan identificar a individuos no hay ningún problema legal. Siempre y cuando los datos sean obtenidos de forma legal, informado a los usuarios de su uso.

# Especificación de Requisitos

## B.1. Introducción

En este apéndice se exponen los requisitos que debe tener el proyecto. Primero se especifican los objetivos generales de la aplicación software y posteriormente se pasa a catalogar todos los requisitos de la aplicación. Por último, se realiza una especificación detallada de los requisitos software de la aplicación.

## B.2. Objetivos generales

Los requisitos de funcionamiento que va a tener este proyecto son los siguientes.

* El objetivo principal que se persigue en este proyecto es crear una aplicación que permita analizar trayectorias. Con este fin se van a desarrollar e implementar las siguientes características.
  + Guardar los datos de la aplicación en una base de datos, estos datos se guardarán ya procesados.
    - Se precisa eliminar errores que pueda haber en las trayectorias debido a fallos aleatorios o de GPS.
    - Detectar cuales, de los puntos de la ruta, un usuario está parado y agrupar los puntos consecutivos en los que se determina que el usuario está parado.
    - Asignar identificadores de OSM a las paradas, estos identificadores corresponden a la dirección en la que determina que está parado.
  + Permitir al usuario decidir exactamente qué datos de la base de datos de la aplicación quiere cargar en el sistema para usar las distintas funcionalidades de la aplicación.
  + Predictor que permita predecir a qué categoría o tipo pertenece la siguiente parada de una trayectoria.
  + Una interfaz amigable para que el usuario pueda utilizar la aplicación.
  + Permitir al usuario que pueda exportar datos de le la base de datos a un fichero CSV.

## B.3. Catálogo de requisitos

A continuación, se enumeran los requisitos funcionales y no funcionales de la aplicación.

### Requisitos funcionales

* **RF1 - Cargar datos CSV en la aplicación**: El usuario debe poder agregar nuevos datos procedentes de ficheros CSV.
  + **RF1.1 – Espera en carga de datos:** El usuario debe ser consciente de que los datos se están cargando en la aplicación y debe esperar.
  + **RF1.2 – Resultado de la carga de datos**: El usuario debe recibir información referente a los datos que se han cargado en la aplicación tras el proceso de carga.
  + **RF1.3 – Exportar resultados de carga:** El usuario podrá exportar los resultados que recibe una vez terminado el proceso de carga de datos.
* **RF2 – Predecir clases de las rutas:** El usuario deberá poder entrenar un clasificador con un conjunto de datos seleccionado, para predecir la clase a la que pertenecerá el siguiente punto de las rutas.
  + **RF2.1 – Selección de datos:** El usuario debe poder elegir que datos desea utilizaren cada momento.
  + **RDF2.2 – Validación cruzada:** El usuario visualizara los resultados de realizar una validación cruzada con el conjunto de datos cargado antes de entrenar el clasificador.
  + **RF2.3 – Datos insuficientes:** El usuario será informado en caso de haber datos insuficientes para entrenar un clasificador, en cuyo caso no podrá entrenar el clasificador con esos datos.
  + **RF2.4 – Reiniciar clasificador:** El usuario podrá reiniciar el clasificador a si como todos los datos que están cargado en la aplicación para ser utilizados.
  + **RF2.5 – Múltiples predicciones:** El usuario podrá utilizar el clasificador múltiples veces para predecir Trayectorias semánticas
  + **RF2.6 – Exportar resultado de predicción:** El usuario podrá exportar los resultados de la predicción obtenidos.
* **RF3 – Exportar datos de la aplicación:** El usuario deberá poder exportar los datos contenidos en la aplicación mediante una sentencia de tipo select.
  + **RF3.1 – Respuesta al usuario:** El usuario debe recibir un aviso si la sentencia select no esta bien formulada o un fichero CSV si la operación fue procesada correctamente.
* **RF4 – Datos de OSM:** El administrador podrá añadir nuevos datos de OSM a la aplicación de Nominatim.

### Requisitos no funcionales

**RNF1 – Facilidad de uso:** El usuario debe recibir en todo momento información sobre los fallos en el uso de la aplicación y el estado de la aplicación

**RNF2 – Información adicional:** El usuario recibirá información adicional en el contexto de la aplicación o facilidades para no necesitar conocer la estructura de la base de datos para poder usarla.

### Limitaciones Funcionales

**LF1 – Usuarios:** La aplicación no deberá ser utilizada por mas de un usuario al mismo tiempo ya que compartirían la misma sesión.

## B.4. Especificación de requisitos

En este apartado se van a especificar los casos de uso, que requisitos funcionales va a abarcar y que característica tiene cada caso de uso.

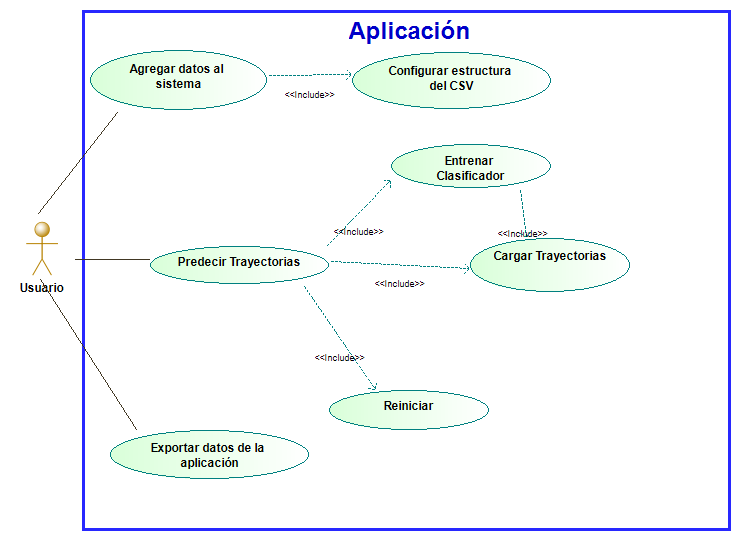
### Actores

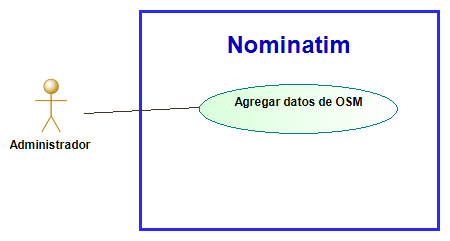
En los casos de uso que se exponen a continuación encontramos dos actores con roles completamente distintos.

**Usuario:** Este es el actor principal de la aplicación y es además el único que interactúa directamente con ella mediante la página web.

**Administrador:** Este es el encargado de que en la aplicación los usuarios tengan los datos necesarios para que Nominatim pueda asociar puntos geográficos con puntos de interés.

### Diagramas de casos de uso





### Especificación de los casos de uso

|  |  |  |
| --- | --- | --- |
| Nombre | CU1 – Agregar datos al sistema | |
| Descripción | El usuario agrega datos al sistema | |
| Requisitos funcionales | RF1, RF1.1, RF1.2, RF1.3 | |
| Usuario | Usuario | |
| Precondiciones | En la página web estar en la ventana de carga, haber rellenado correctamente la configuración del CSV | |
| Secuencia Normal | Paso | Acción |
| 1 | El usuario solicita cargar los datos |
| 2 | El navegador carga los datos en el servidor |
| 3 | El navegador pide la página cada 5 segundos al servido, si termina el proceso de carga, varia la página y salta al paso 7 |
| 4 | El servidor lee los archivos con extensión especificada |
| 5 | El servidor procesa los datos |
| 6 | El servido guarda los resultados en la base de datos |
| 7 | Se muestra la información de los resultados y habilita un botón para que el usuario pueda exportar el resultado. |
| Excepciones | Paso | Acción |
| 1 | Se produce un error al intentar cargar los datos |
| 2 | El servidor envía una ventana de error al navegador |
| Postcondiciones | Se añaden datos a la base de datos | |
| Frecuencia | Baja | |
| Importancia | Critica | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU1.1 – Configurar estructura del CSV | |
| Descripción | El usuario configura el lector de CSV | |
| Requisitos funcionales | RF1 | |
| Usuario | Usuario | |
| Precondiciones | Estar en la ventana de carga de datos | |
| Secuencia Normal | Paso | Acción |
| 1 | El usuario rellena los campos |
| 2 | El usuario pulsara el botón de cargar |
| Excepciones | Paso | Acción |
| 1 | Al pulsar agregar hay campos obligatorios sin rellenar o campos inválidos |
| 2 | El navegador informa al usuario del primer campo vacío o invalido |
| Postcondiciones | Permite cargar los datos en el servidor | |
| Frecuencia | Baja | |
| Importancia | Critica | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU2 – Predecir Trayectorias | |
| Descripción | Permite al usuario predecir la clase a la que pertenecerá la próxima ubicación en una trayectoria | |
| Requisitos funcionales | RF2, RF2.5, RF2.6 | |
| Usuario | Usuario | |
| Precondiciones | Tener el clasificador entrenado, tener datos cargados | |
| Secuencia Normal | Paso | Acción |
| 1 | El usuario pulsa el botón de predecir |
| 2 | El servidor pasa los datos actualmente cargados por el clasificador |
|  | 3 | El navegador muestra en una tabla los resultados y permite exportarla |
|  | 4 | El usuario puede exportar la tabla y/o volver a predecir o cargar datos |
| Excepciones | Paso | Acción |
|  |  |
| Postcondiciones |  | |
| Frecuencia | Alta | |
| Importancia | Critica | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU2.1 – Entrenar clasificador | |
| Descripción | Entrena de forma automática un clasificador, realiza la validación cruzada | |
| Requisitos funcionales | RF2, RF2.5 | |
| Usuario | Usuario | |
| Precondiciones | Terminar de cargar los datos en el sistema, no hay clasificador entrenado | |
| Secuencia Normal | Paso | Acción |
| 1 | El servidor realiza una validación cruzada |
| 2 | Entrena un clasificador |
| 3 | Presenta los resultados de la validación en el navegador |
| Excepciones | Paso | Acción |
|  |  |
| Postcondiciones | Obtenemos un clasificador entrenado | |
| Frecuencia | Alta | |
| Importancia | Critica | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU2.2 - Reiniciar | |
| Descripción | Devuelve a todos los componentes de predicción a su estado inicial | |
| Requisitos funcionales | RF2.4 | |
| Usuario | Usuario | |
| Precondiciones | Estar en la ventana de predicciones | |
| Secuencia Normal | Paso | Acción |
| 1 | El usuario pulsa el botón de reiniciar |
| 2 | El servido devuelve las variables de predicción a su estado original |
| 3 | Se refresca la página web |
| Excepciones | Paso | Acción |
|  |  |
| Postcondiciones | Desaparecen todo lo realizado en la ventana predicciones | |
| Frecuencia | Media | |
| Importancia | Alta | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU2.3 – Cargar Trayectorias | |
| Descripción | Carga trayectorias en memoria | |
| Requisitos funcionales | RF2.1, RF2.3 | |
| Usuario | Usuario | |
| Precondiciones | Estar en la ventana de predicción | |
| Secuencia Normal | Paso | Acción |
| 1 | El usuario rellena o no los campos de la columna cargar datos |
| 2 | El usuario presiona el botón de cargar |
|  | 3 | Se envía una solicitud al servidor |
|  | 4 | El servidor almacena los datos en memoria |
| Excepciones | Paso | Acción |
| 1 | Si hay campos incompatibles con la consulta select devuelve un error |
| Postcondiciones | Datos cargados en el sistema | |
| Frecuencia | Alta | |
| Importancia | Critica | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU3 – Exportar datos de la aplicación | |
| Descripción | Permite a un usuario exportar datos almacenados en la base de datos | |
| Requisitos funcionales | RF3, RF3.1 | |
| Usuario | Usuario | |
| Precondiciones | Estar en la ventana exportar, que el usuario tenga conocimientos de SQL básicos | |
| Secuencia Normal | Paso | Acción |
| 1 | El usuario rellena la consulta select |
|  | 2 | El usuario presiona el botón para hacer la exportación |
|  | 3 | El servidor detecta si la consulta pide datos geoespaciales, en caso afirmativo ir al paso 5 |
|  | 4 | El servidor hace una petición normal a la base de datos saltar al paso 7 |
|  | 5 | El servidor realiza una consulta con datos geoespaciales |
|  | 6 | El servidor convierte la columna de datos geoespacial en una de longitud y otra de latitud |
|  | 7 | Se devuelve un archivo CSV con los datos |
| Excepciones | Paso | Acción |
| 1 | Si la consulta es invalida el servidor devuelve una notificación |
| Postcondiciones |  | |
| Frecuencia | Baja | |
| Importancia | Media | |

|  |  |  |
| --- | --- | --- |
| Nombre | CU4 – Agregar datos de OSM | |
| Descripción | Añadir datos a Nominatim | |
| Requisitos funcionales | RF4 | |
| Usuario | Administrador | |
| Precondiciones | Estar logrado con el usuario de Nominatim en el servidor, tener internet | |
| Secuencia Normal | Paso | Acción |
| 1 | El administrador debe descargar los nuevos datos de internet |
| 2 | El administrador debe pasar estos datos al setup de Nominatim |
| Excepciones | Paso | Acción |
|  |  |
| Postcondiciones | Nuevas zonas para obtener puntos de interés | |
| Frecuencia | Baja | |
| Importancia | Critica | |

# Especificación de diseño

## C.1. Introducción

En este apartado se esponen los aspectos de diseño mas importantes de la aplicacion

## C.2. Diseño de datos

## C.3. Diseño procedimental

## C.4. Diseño arquitectónico

# Documentación técnica de programación

## D.1.Introducción

## D.2. Estructura de directorios

## D.3. Manual del programador

## D.4. Compilación, instalación y ejecución del proyecto

## D.5. Pruebas del sistema

# Documentación de usuario

## E.1. Introducción

## E.2. Requisitos de usuarios

## E.3. Instalación

## E.4. Manual del usuario