**Distributed Systems Programming 600089**

Distributed Systems API: Report

Student Number: 201601628

Submitted April 2020

Words: 1000

# 1. APIs

**Outline what an API is and does**

An Application Programming Interface (API) exposes functionality of one system to others. In this sense, APIs are ‘consumed’ by other applications, serving as an interface between two systems. They allow for a developer to integrate other systems or interact with them without needing to know implementation details; they need only know what is available (e.g. endpoints). This allows for systems to be decoupled and distributed.

APIs are standard contracts that define how developers communicate with a service, and the kind of output those developers should expect to receive back.

One of many examples is the Google Maps API, which allows developers to integrate features without having to develop or manage their own maps system. Apple improves the camera API, all the apps that rely on it will take advantage of that improvement automatically. If APIs didn’t exist to make this easy, app developers would have to create their own camera software and interpret the camera hardware’s inputs. But Apple’s operating system developers have done all this hard work so the developers can just use the camera API to embed a camera, and then get on with building their app Google exposes APIs like this to web developers, who can then use the APIs to plop complex objects right on their website. If APIs like this didn’t exist, developers might have to create their own maps and provide their own map data just to put a little interactive map on a website.

security

APIs are also used to control access to hardware devices and software functions that an application may not necessarily have permission to use. That’s why APIs often play a big role in security.

expose this information via an API because it’s possible to control access to it.

**How it manages request**

RESTful APIs describe a popular architecture that uses relatively simple HTTP requests to defined endpoints. … A RESTful API

**Discuss why this API is stateless and describe the difference between a stateless and stateful server.**

A server may be described as stateful or stateless. The former dictates that the server stores session information in regard to the requester, between requests. This may, for example, monitor whether the user is logged in to the server and be used to authenticate multiple requests. On the contrary, a stateless server does not store information about the current user and each request is a single, self-contained message. Using the RESTful API developed for this project as an example, requests are authenticated by passing an API key in the header of the request.

# 2. Route Mapping in WebAPI

WebAPI features multiple schemes for defining endpoints.

Action, <method name>,

Briefly explain what route mapping is

how WebAPI uses the id parameter

what actions are.

<https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/routing-in-aspnet-web-api>

# 3. HTTP Requests

Briefly outline what GET, POST and DELETE requests areprovide screenshots of where you have used these requests in your server project to illustrate your written work.

# 4. API Keys

In the created API, methods requiring an authenticated user expect to receive an API key in the header of the request, which is then searched for in the database for verification. Briefly describe how your Server and Client use the API key. Identify if you think an API key is a good or bad option for identifying users, giving your reasons. Is the API key safe in this project? How would you ensure this API key was kept safe if you were developing this Server/Client in the ‘real world’?

# 5. The RSA Algorithm

Outline the steps in the RSA algorithm. IN BULLET POINTS

<https://www.tutorialspoint.com/cryptography_with_python/cryptography_with_python_understanding_rsa_algorithm.htm>

### **Step 1: Generate the RSA modulus**

The initial procedure begins with selection of two prime numbers namely p and q, and then calculating their product N, as shown −

N=p\*q

Here, let N be the specified large number.

### **Step 2: Derived Number (e)**

Consider number e as a derived number which should be greater than 1 and less than (p-1) and (q-1). The primary condition will be that there should be no common factor of (p-1) and (q-1) except 1

### **Step 3: Public key**

The specified pair of numbers **n** and **e** forms the RSA public key and it is made public.

### **Step 4: Private Key**

Private Key **d** is calculated from the numbers p, q and e. The mathematical relationship between the numbers is as follows −

ed = 1 mod (p-1) (q-1)

The above formula is the basic formula for Extended Euclidean Algorithm, which takes p and q as the input parameters.

## **Encryption Formula**

Consider a sender who sends the plain text message to someone whose public key is **(n,e).** To encrypt the plain text message in the given scenario, use the following syntax −

C = Pe mod n

## **Decryption Formula**

The decryption process is very straightforward and includes analytics for calculation in a systematic approach. Considering receiver**C** has the private key **d**, the result modulus will be calculated as −

Plaintext = Cd mod n

# 6. The AES Algorithm

Outline the steps in the AES algorithm. IN BULLET POINTS

<https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm>

It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix −

the number of rounds in AES is variable and depends on the length of the key

The schematic of AES structure is given in the following illustration −



## **Encryption Process**

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below −



### **Byte Substitution (SubBytes)**

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

### **Shiftrows**

Each of the four rows of the matrix is shifted to the left. Any entries that ‘fall off’ are re-inserted on the right side of row. Shift is carried out as follows −

* First row is not shifted.
* Second row is shifted one (byte) position to the left.
* Third row is shifted two positions to the left.
* Fourth row is shifted three positions to the left.
* The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

### **MixColumns**

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

### **Addroundkey**

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

## **Decryption Process**

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order −

* Add round key
* Mix columns
* Shift rows
* Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related.

# 7. Entity Framework

Briefly describe what the Entity Framework is and what it does.

<https://www.entityframeworktutorial.net/what-is-entityframework.aspx>

Entity Framework is an open-source [ORM framework](https://en.wikipedia.org/wiki/Object-relational_mapping) for .NET applications supported by Microsoft. It enables developers to work with data using objects of domain specific classes without focusing on the underlying database tables and columns where this data is stored. With the Entity Framework, developers can work at a higher level of abstraction when they deal with data, and can create and maintain data-oriented applications with less code compared with traditional applications.

Official Definition: “Entity Framework is an object-relational mapper (O/RM) that enables .NET developers to work with a database using .NET objects. It eliminates the need for most of the data-access code that developers usually need to write.”

As per the above figure, Entity Framework fits between the business entities (domain classes) and the database. It saves data stored in the properties of business entities and also retrieves data from the database and converts it to business entities objects automatically.

## **Entity Framework Features**

* **Cross-platform:** EF Core is a cross-platform framework which can run on Windows, Linux and Mac.
* **Modelling:** EF (Entity Framework) creates an EDM (Entity Data Model) based on POCO (Plain Old CLR Object) entities with get/set properties of different data types. It uses this model when querying or saving entity data to the underlying database.
* **Querying:** EF allows us to use LINQ queries (C#/VB.NET) to retrieve data from the underlying database. The database provider will translate this LINQ queries to the database-specific query language (e.g. SQL for a relational database). EF also allows us to execute raw SQL queries directly to the database.
* **Change Tracking:** EF keeps track of changes occurred to instances of your entities (Property values) which need to be submitted to the database.
* **Saving:** EF executes INSERT, UPDATE, and DELETE commands to the database based on the changes occurred to your entities when you call the SaveChanges() method. EF also provides the asynchronous SaveChangesAsync() method.
* **Concurrency:** EF uses Optimistic Concurrency by default to protect overwriting changes made by another user since data was fetched from the database.
* **Transactions:** EF performs automatic transaction management while querying or saving data. It also provides options to customize transaction management.
* **Caching:** EF includes first level of caching out of the box. So, repeated querying will return data from the cache instead of hitting the database.
* **Built-in Conventions:** EF follows conventions over the configuration programming pattern, and includes a set of default rules which automatically configure the EF model.
* **Configurations:** EF allows us to configure the EF model by using data annotation attributes or Fluent API to override default conventions.
* **Migrations:** EF provides a set of migration commands that can be executed on the NuGet Package Manager Console or the Command Line Interface to create or manage underlying database Schema.

Compare code first, model first and database first techniques and describe what a migration is/does.

There are several documented approaches to developing applications with Entity Framework:

* Code First
* Model First
* Database First

The suitability of each approach largely depends on the scope of the project and the information available at the time of development.

The code first method allows the developer to focus on development of backend code without concern for database design, which is instead managed by Entity Framework. Through a variety of decorators and conventions (e.g. “[Key]” or PrimaryKeyId) and the manual creation of “context” simply to define the required collections to be stored, EF can generate any number of SQL databases and, likewise, tables within them.

On the other hand, the model first approach requires that…

Finally, in database first…

# 8. Reflections

Finally, write a short reflective statement about which tasks you completed and to what level, any problems you had with any of the functionality and how you overcame these problems (if you managed to).

All tasks were completed, with the largest difficulty being interpretation of requirements. For example, the API might be improved by incorporating hashing into the