**Software Engineering: Assignment # 1**

**Documentation for Automated Air Traffic Control System**

­­BSCE20033 – M. Irtaza

BSCE20016 – Muneeb Sami

BSCE20007 – M. Abdullah

**Purpose and Functionality**

The main purpose of this software development project is to create a user-friendly system that controls the flow of air traffic around airports in order to increase air traffic efficiency and safety. Due to several disasters caused by human error, it was felt that air traffic controllers should to be automated. To ensure effective communication between the pilot and the control room operator, the system must be stable. There must be a system-wide emergency protocol that would maintain contact between the pilot and the control room operator in the event of an emergency. To that end, there should be a backup system. Cross-departmental coordination between the crisis management team and other departments is essential for expediting response and minimizing damage and loss of life.

**Intended Users**

The air traffic controller and Arrival/Departure Admin are among the users we are mainly considering. The system should offer the ATC the essential features related to daily operations. Separating aircraft to avoid collisions, streamlining and organizing traffic, and, when possible, offering information and other help to pilots are all examples of this.

The arrival and departure manager and the air traffic controller are the users who will utilise the system the most regularly. The flight administrator, the system administrator, the system analyst, the software tester, etc. are examples of additional users.

**Functional Requirements**

* Every flight is in communication with our system.
* The Metrological Section regularly upgrades the system.
* The system determines the aircraft's present location. The user can observe a visual representation of the aircraft's current position via a graphical user interface (GUI)
* A flight is only permitted to take off when the weather is favourable.
* In the event of a crisis, an emergency protocol should be followed.
* Determines a scheduled departure flow to preserve an ideal throughput at the runway, which will lower waiting times at holding areas, and relays the information to other airport stakeholders
* Calculates a planned Arrival flow with the intention of preserving the best throughput at the runway, lowering arrival queuing, and communicates the data to various stakeholders.
* When an aircraft looks to be flying too close to the ground or is headed for a piece of terrain, the Minimum Safe Altitude Warning (MSAW) tool warns the controller.
* Short Term Conflict Alert (STCA), which informs the controller before the loss of separation by examining potential conflicting trajectories over a time period of around 2 or 3 minutes.
* Allows controllers and pilots to send digital messages instead of using radiotelephony.

**Non-functional Requirements**

* Good performance is expected from the software. The hardware must be compatible with all of the AATC software's subsystems in order to guarantee smooth operation of the software. The system should be created and implemented in a way that allows it to effectively manage failures and exceptions. There must be a backup mechanism to resume operations quickly in the event of a system failure.
* The system must be secure. Only the authorized users should be able to use it. It should be resistant to security attacks such that any attack from a third party e.g. virus shouldn't impact the performance of the software.
* Each employee should have a log in. The software should record the user's login time, activity (such as which flight was managed at which time, which answer was made in response to a flight request, which details were sent to the flight pilot, etc.) on the software, and logout time. The software should inaccessible to anyone without authentication.
* The system should be accurate and reliable such that all the readings are precise.
* The software should be portable from hardware to hardware, so there is no hindrance in the event of hardware upgrades.
* It should be supported by all the widely used operating systems (Windows, Linux, MacOS) and different distributions of the operating systems.
* The software should be easy to maintain. The whole code should be easily understandable. Updating the software should not be cumbersome.