**Duration: 5 Days**

**Pre-requisites**

1. No prior knowledge of the Elastic Stack is required. This course is designed for individuals new to the Elastic Stack Comfort using the terminal or command line is recommended

**Requirements**

1. Participants must use their own desktop or laptop system
2. Internet connection capable of streaming video
3. Windows 7 or later
4. Java version 1.8u20 or later installed
5. A modern web browser
6. At least 20% free disk space
7. **Basic flow of data in Elasticsearch**
   * what is Elasticsearch and typical use-cases
   * shards and replicas; packaging
   * installation; configuration files
   * indexing; what is an index, type and ID
   * mappings; stored and indexed fields; \_source and \_all
   * analysis basics
   * realtime get
   * search; how searches are distributed to shards
   * ranking by TF/IDF and BM25
   * aggregations and doc values introduction
   * updates; versioning
   * deletes; introduction to Lucene segment merges
   * **Lab**
     + CRUD operations
     + query and filter
     + pagination

1. **Controlling how data is indexed and stored**
   * mappings and mapping types
   * multi-field definitions
   * default mappings; dynamic mappings
   * texts, keywords, integers and other core types
   * date formats
   * predefined fields; when to store fields separately vs using \_source
   * analyzers; using the Analyze API
   * char filters
   * tokenizers: standard vs whitespace
   * token filters: lowercase, stopwords, synonyms, ngrams and shingles
   * **Lab**
     + exact match vs full-text search
     + using the asciifolding token filter for better internationalization
     + using language analyzers to support stemming

1. **Searching through your data**
   * selecting fields, source filtering and fielddata fields
   * sorting and pagination
   * search basics: term, range and bool queries
   * enable caching through the filter context
   * match query: configuring the analyzer, operator, common terms and fuzziness
   * query string and simple query string queries
   * **Lab**
     + using various ways of selecting fields
     + configure sorting and pagination
     + using a bool query to combine different match, range and term queries
     + boosting exact matches above stemmed ones

1. **Aggregations**
   * relationships between queries and aggregations; post filter, global aggregations
   * general optimizations: avoid script fields, set result size to 0 to cache
   * metrics aggregations: stats, cardinality, percentiles
   * why terms, cardinality and percentiles are approximate
   * multi-bucket aggregations: terms, ranges and histograms
   * single-bucket aggregations and nesting; how nesting works
   * **Lab**
     + configure sizes of results, per-shard and overall buckets
     + computing the cardinality of a field
     + sorting buckets by results of sub-aggregations
     + optimizing terms queries by configuring collect mode
     + nest the sum and histogram aggregations
2. **Data visualization through Kibana**
   * installation and configuration
   * index patterns; refreshing the fields list
   * discovering and searching raw data
   * Lucene query syntax
   * visualizing data; types of visualizations and their use
   * Timelion charts; using the Timelion query language
   * building dashboards
   * **Lab**
     + building complex queries through the Lucene query syntax
     + digging deeper into data through sub-aggregations
     + building dashboards on top of saved searches and visualizations
     + comparing different data series in Timelion (raw average vs moving average)
3. **Data ingestion through Logstash**
   * installation
   * inputs: popular input plugins and their configuration options
   * codecs: parsing JSON and multiline logs
   * filters: using grok and geoip to parse and enrich data
   * outputs: popular output plugins and their options
   * pipeline pattern: using Logstash on every logging box
   * using Logstash with Kafka and Redis as a buffer
   * adjusting pipeline workers and batch sizes
   * adjusting Logstash heap size
   * specific plugin tunables
   * **Lab**
     + configuring Logstash to parse and enrich Apache logs
     + tuning Logstash for throughput
     + using Logstash with Kafka
4. **Data collection using Beats**
   * installation: Packetbeat, Topbeat, Filebeat
   * Filebeat tunables
   * parsing JSON logs
   * sending logs directly to Elasticsearch
   * using Ingest nodes
   * sending logs directly to Logstash
   * sending logs to Logstash via Kafka
   * **Lab**
     + setting up TopBeat to push metrics to Elasticsearch
     + shipping parsing Apache logs via Filebeat and Ingest node
     + shipping and parsing Apache logs via Filebeat and Logstash
5. **Data collection using rsyslog**
   * installation
   * plugins: main input modules and their configurations
   * message modifiers: using mmnormalize to parse unstructured data in a scalable way
   * parsing JSON logs
   * using grok in rsyslog
   * tuning queues, workers and batch sizes
   * Rainerscript: variables, conditionals, loops and lookup tables
   * using rulesets to manage multiple data flows
   * writing data to Elasticsearch
   * coupling rsyslog with Logstash via Redis/Kafka
   * **Lab**
     + sending local syslog to Elasticsearch
     + tailing files with rsyslog and sending them to Kafka
     + using rulesets to separate local and remote logs
     + parsing logs with mmnormalize and sending them to Elasticsearch
6. **Data collection using Logagent-js**
   * installation
   * running on-demand or as a service
   * parsing rules
   * GeoIP matching and database updates
   * UDP syslog and other listeners
   * **Lab**
     + parsing and sending local Apache and syslog to Elasticsearch
     + build a pipeline from rsyslog to Elasticsearch through Logagent
7. **Relevancy tuning**
   * analysis: stopwords, synonyms, ngrams and shingles and their alternatives
   * using the Reindex API when mappings need to be changed
   * a deep look into BM25
   * multi-match query: choosing between best fields, most fields and cross fields modes
   * tweaking the score with the function score query
   * **Lab**
     + using the letter tokenizer as an option for URL matching
     + using ngrams to tolerate typos
     + using shingles to match compound words
     + implement hashtag search via the word delimiter token filter
     + searching across multiple fields
     + boosting documents based on date and number of views
     + typo tolerance without using ngrams
     + reducing the impact of common words without using stopwords
8. **Advanced aggregations**
   * finding trends and outliers with the significant terms aggregation
   * cheaper and more representative results with the sampler aggregation
   * field collapsing with the top hits aggregations
   * pipeline aggregations; moving averages
   * **Lab**
     + checking trends the significant terms aggregation
     + show the latest hit per category
     + using the moving average aggregation
9. **Working with relational data**
   * arrays and objects; why the offer the best performance and when they fail
   * nested documents
   * nested queries; using inner hits
   * parent-child relations
   * denormalizing and application-side joins
   * deciding on which feature/technique to use
   * **Lab**
     + model a one-to-one relationship
     + model a query-heavy one-to-many relationship
     + model an update-heavy one-to-many relationship
     + model a many-to-many relationship
10. **Percolator**
    * percolator basics
    * configuring mappings for percolation
    * using routing, filters, sorting and aggregations with the Percolator Query
    * **Lab**
      + using Percolator to trigger alerts
      + using metadata to filter and aggregate matching queries
11. **Suggesters**
    * overview of types and requests
    * term vs. phrase suggester
    * how the phrase suggester collects candidates
    * using a shingle field to score candidate phrases
    * completion vs context suggesters
    * completion suggesters vs prefix queries
    * mapping for completion suggesters
    * weights and fuzzy matches
    * payloads for instant-search kind of autocomplete
    * **Lab**
      + using the term suggester to suggest single word corrections
      + using the phrase suggester against a shingle field for multi-word suggestions
      + using a separate index for autocomplete
      + using the \_suggest endpoint instead of \_search
      + boosting suggestions via static weights
      + add fuzzy support for suggestions
      + filtering suggestions
      + using metadata for ranking suggestions (terms, location)
12. **Geo-spatial search**
    * Basics: geo-point and geo-shape types
    * how shape matching is done via geohashes
    * distance, distance range and bounding box queries
    * **Lab**
      + indexing geo-points and searching them via bounding box and polygon queries
      + filtering and aggregating geo-points by distance
      + matching a shape against a point
13. **Highlighting**
    * how the default highlighter works
    * common highlighter options: size, order and number of fragments
    * postings highlighter: overhead, use-cases, mapping
    * fast vector highlighter: using term vectors for extra flexibility
    * **Lab**
      + selecting fields to highlight and disabling \_source from the response
      + choosing highlight tags, number of fragments, their size and order
      + using the postings highlighter for long natural language fields
      + using the fast vector highlighter for multi-fields