**Introduction**

This document is relative with Neo4j program about timetable of GMIT. It can describe all relationship like courses, room, tutors and others.

**What to store**

This graph contains courses, rooms and tutors. Each of them have relationship with other.

Course has name and available time attributes and relates with room and tutor

Room has number and seats attributes also relates with course and tutor

Tutor has name and relate with course and room

**About Noe4j**

**Neo4j** is a [graph database](https://en.wikipedia.org/wiki/Graph_database) management system developed by Neo Technology, Inc. Described by its developers as an [ACID](https://en.wikipedia.org/wiki/ACID)-compliant transactional database with native graph storage and processing,[[3]](https://en.wikipedia.org/wiki/Neo4j#cite_note-product-details-3)Neo4j is the most popular graph database according to db-engines.com.[[4]](https://en.wikipedia.org/wiki/Neo4j#cite_note-DB-Engines-4)

Neo4j is available in a [GPL3](https://en.wikipedia.org/wiki/GNU_General_Public_License)-licensed [open-source](https://en.wikipedia.org/wiki/Open-source_software) "community edition", with [online backup](https://en.wikipedia.org/wiki/Remote_backup_service) and [high availability](https://en.wikipedia.org/wiki/High_availability) extensions licensed under the terms of the [Affero General Public License](https://en.wikipedia.org/wiki/Affero_General_Public_License" \o "Affero General Public License). Neo also licenses Neo4j with these extensions under closed-source commercial terms.[[5]](https://en.wikipedia.org/wiki/Neo4j#cite_note-LICENSE-5)

Neo4j is implemented in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and accessible from software written in other languages using the [Cypher Query Language](https://en.wikipedia.org/wiki/Cypher_Query_Language) through a transactional HTTP endpoint, or through the binary 'bolt' protocol.

**How to store**

To store data into neo4j, it should use nodes and relationships to connect them

Node: user create a node and give it several attributes to specific and connect to other nodes

By using: create (xxx:Xxx{attribute1:’’, attribute2:’’})

Relationship: this is a link between two nodes to display their relationship