Cairo University  
Faculty of Computers and Information



**CS251**

**Software Engineering I**

SRS 13

Software Design

Gobran Asker

Abdulelah Adam

Ali Gad

December & 2019

Contents

[Instructions [To be removed] 3](#_Toc25570060)

[Team 3](#_Toc25570061)

[Document Purpose and Audience 3](#_Toc25570062)

[System Models 3](#_Toc25570063)

[I. System Decomposition 3](#_Toc25570064)

[II. Class diagrams 6](#_Toc25570065)

[Important Algorithm 7](#_Toc25570066)

[III. Sequence diagrams 7](#_Toc25570067)

[Class - Sequence Usage Table 9](#_Toc25570068)

[Ownership Report 10](#_Toc25570069)

[Policy Regarding Plagiarism: 10](#_Toc25570070)

[References 10](#_Toc25570071)

[Authors 10](#_Toc25570072)

# Instructions [To be removed]

* **IMPORTANT. Rename this document to CS251-TAName-LeaderID-SDDocument.docx**

**(e.g. CS251-MohamedSamir-20040752-SDDDocument.docx)**

* **Remove the following notes and any red notes**
* **This document is the template document for your Software Design.**
* **For further guidelines and information, READ project details document (C251-SE2014-Project Description-v2.0).**

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20160389 | Gobran Yehia Asker | [gobran@stud.fci-cu.edu.eg](mailto:gobran@stud.fci-cu.edu.eg) |  |
| 20170416 | Abdulelah Adam Abdi | [Abdulelah859@gmail.com](mailto:Abdulelah859@gmail.com) | 01095075448 |
| 20170362 | Ali Ahmed Ahmed Gad | [Aligad204@gmail.com](mailto:Aligad204@gmail.com) |  |

# Document Purpose and Audience

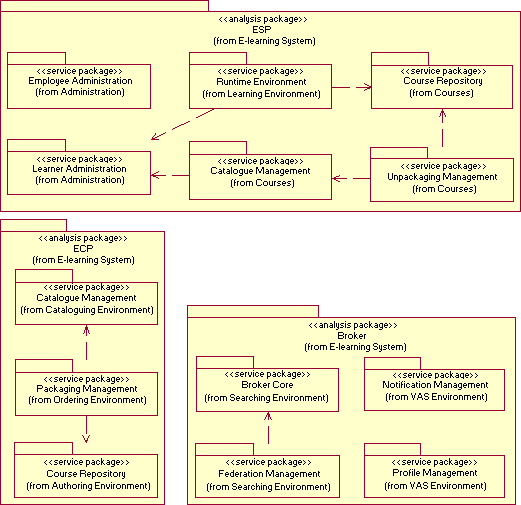
* **What's this Document?**

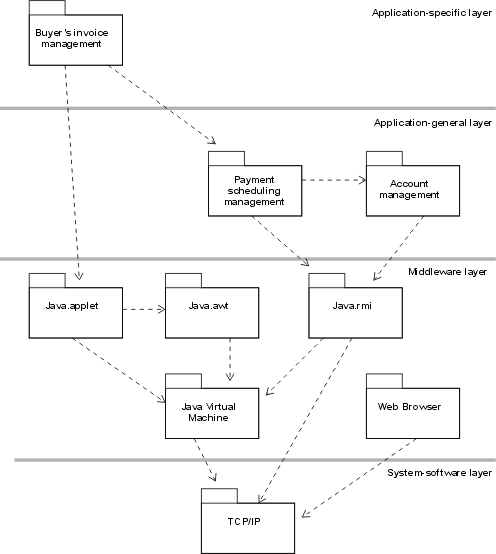
# - This document is to present a detailed description to the related readers of this document about "Social Network back-end API". It will explain the purpose and features of the system, its interfaces, all functionalities of the system, constraints and restrictions that's agreed among stakeholders and developers and How the system actors interact with the system, and It will be proposed to the product owner for its approval

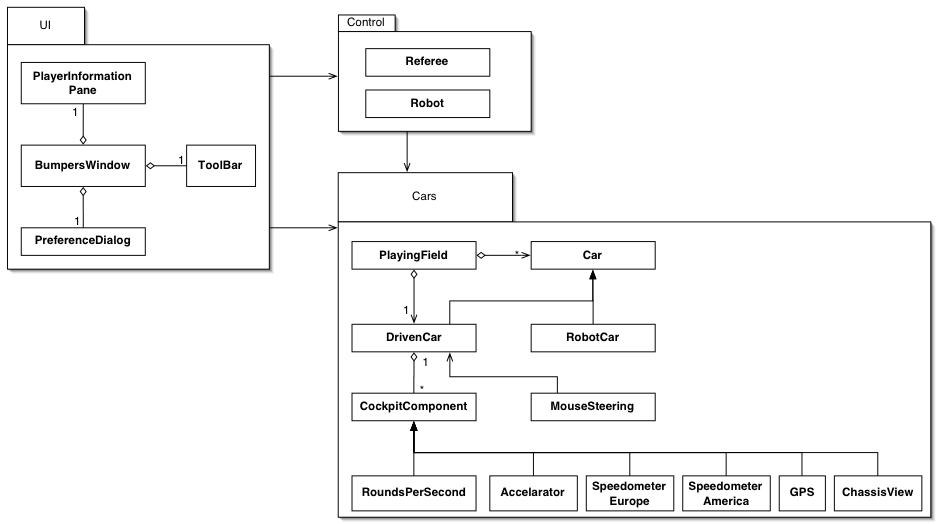
# System Models

## I. System Decomposition

* **Provide and describe a figure that depicts the overall system decomposition.**
* **Read about “Software Coupling and Cohesion” Concepts. Reflect that in model**
* **The three following images from web are for purpose of explaining. First one is how model is designed, but there are little connections (so it is not so good).The other two ones have better connection. Consider the 3 to know how a good model should be.**
* **Remove these images from your report and draw your own system decomposition.**







## II. Class diagrams

* **You should provide your class diagram. In case on diagram is so complex, divide it to several ones of reasonable size or draw separate ones, each for one of the components on the system decomposition diagram.**
* **Class diagram is a static diagram and should not represent any dynamic flow of events.**
* **Put stereotypes of the classes to give more information. UML predefines some stereotypes like: <<interface>>, <<type>>, <<implementationClass>>, <<enumeration>>, etc. and you create your own also.**
* **Put Relationships between classes and the types of the relationships.**
* **Put multiplicity.**
* **Put relationship name (e.g. faculty "offer" course).**
* **Put attributes in the classes.**
* **Put functions & Put parameters.**
* **Put data types of each attributes and the parameters.**
* **Make sure to include all domain (entity), boundary and control classes needed to implement the system.**
* **Highly perfered: Each class has a corresponding interface**
  + **Let all objects parameters and returns be of interface type.**
  + **See Shopping Cart Case Study**
* **Following is Shopping Cart Component class diagram.**



**List down your classes and describe them**

| **Class ID** | **Class Name** | **Subsystem ID** | **Description & Responsibility** |
| --- | --- | --- | --- |
|  |  |  |  |

* **In the above table make sure that each class belongs to a subsystem.**
* **In the above table ALL classes should belong to subsystems. And each subsystem should at least contain one class.**

### Important Algorithm

* **If any method in a class is implementing an algorithm (complex enough), then you should describe it here.**

## III. Sequence diagrams

* **List Sequence diagrams for all requirements. Provide for each Sequence an ID.**
* **Usually each use case is represented by a sequence diagram or more.**
* **Overall, all the diagrams should represent all requirements and possible flows.**
* **Make sure that each object in the sequence diagram has a corresponding class in the class description table above. If not, it will be REJECTED.**
* **Put actual function calls with proper parameters and return types corresponding to class diagrams.**
* **Following are couple of examples for small / meduim examples. We expect such diagrams, however there is a missing thing in them. Most of calls don’t have parameters. Please always specify the parameters in the call, matching the class diagram.**





### Class - Sequence Usage Table

* **In this table, we will list EVERY class in class diagram and which sequences used this class diagram. This helps in avoiding either unused classes or extra classes appears in sequence diagrams. In "Overall used methods" section, put all functions appeared in all sequences. If this table was built in ignorance of actual class / sequence diagrams = REJECTED for whole document.**

| **Class Name** | **Sequence Diagrams** | **Overall used methods** |
| --- | --- | --- |
| E.g. Employee | 1, 3, 5 (means Seq Ids 1, 3, 5 used Employee class) | Save, GetData |
|  |  |  |

# Ownership Report

* **Remove the following notes and any red notes**
* **For every item in this document, write the owners. If someone is owner of something, s/he understands it 100.%**
* **Team leader must verify the table with the team members.**

|  |  |
| --- | --- |
| **Item** | **Owners** |
|  |  |
|  |  |

# Policy Regarding Plagiarism:

**Students have collective ownership and responsibility of their project. Any violation of academic honesty will have severe consequences and punishment for ALL team members.**

1. تشجع الكلية على مناقشة الأفكار و تبادل المعلومات و مناقشات الطلاب حيث يعتبر هذا جوهريا لعملية تعليمية سليمة
2. ساعد زملاءك على قدر ما تستطيع و حل لهم مشاكلهم فى الكود و لكن تبادل الحلول غير مقبول و يعتبر غشا.
3. أى حل يتشابه مع أى حل آخر بدرجة تقطع بأنهما منقولان من نفس المصدر سيعتبر أن صاحبيهما قد قاما بالغش.
4. قد توجد على النت برامج مشابهة لما نكتبه هنا أى نسخ من على النت يعتبر غشا يحاسب عليه صاحبه.
5. إذا لم تكن متأكدا أن فعلا ما يعد غشا فلتسأل المعيد أو أستاذ المادة.
6. فى حالة ثبوت الغش سيأخذ الطالب سالب درجة المسألة ، و فى حالة تكرار الغش سيرسب الطالب فى المقرر.

# References

* http://www.mhhe.com/engcs/compsci/pressman/graphics/Pressman5sepa/common/cs1/design.pdf

# Author(s)

* Mostafa Saad and Mohammad El-Ramly