Project Specification

Renata Guizzardi

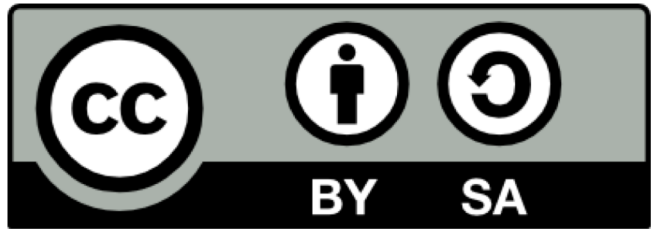
renata.guizzardi@unitn.it

DISI/unitn

Licenza d'uso e distribuzione

- Quest'opera è concessa in licenza Creative.
  Commons Attribuzione-CondivisioneCome 4.0 Internazionale;
- Siete liberi di farlo (per qualsiasi scopo, anche commerciale):
  - Condividi: copia e ridistribuisci il materiale in qualsiasi momento medio o formato;
  - Adattare: remixare, trasformare e costruire sul materiale;
- Alle seguenti condizioni:
  - Attribuzione: è necessario dare il credito appropriato, fornire un link alla licenza e indicare se sono state apportate modifiche. Potete farlo in qualsiasi modo ragionevole, ma non in alcun modo che suggerisca che il licenziante approvi voi o il vostro uso;
  - ShareAlike: se remixate, trasformate o costruite sul materiale, dovete distribuire i vostri contributi sotto la stessa licenza dell'originale.

Ulteriori informazioni sono disponibili al seguente indirizzo:
http://creativecommons.org/licenses/by-sa/4.0/



Obiettivi del Progetto

- Catturare i requisiti di sistema utilizzando *user stories.*
- Seguire un metodo di sviluppo agile.
- Strutturare il sistema in moduli da esporre tramite APIs.
- Gestire il codice sorgente in un ambiente collaborativo con controllo di versione.

Attività del Progetto

- Trovare un cliente e definire che sistema il gruppo fara.
- Definizioni di *user stories* insieme al cliente e creazione di un *Product Backlog* (mantenuti su GitHub).
- Pianificazioni iterativa di ogni *Sprint*.
- Esecuzione iterativa di ogni Sprint per implementare APIs. Le APIs devono essere *versioned*, perché potranno cambiare durante il progetto.
- Descrizione del processo SCRUM utilizzato, con i piani degli Sprint inclusi.

Note

- Gruppi: 5 persone, idealmente.
- Il gruppo si divide internamente il lavoro di implementazione e di *testing*, utilizzando SCRUM. Comunque, tutti membri devono avere *commits* visibili su GitHub.
- Ogni gruppo ha un solo GitHub. **OGNI** membro del gruppo codice e la cosa deve trasparire da *commit* fatti dallo studente visibili su GitHub.
- The **«How to demonstrate»** field of the implemented user stories should be linked to the **documentation of the service** which implements it.

Wiki

- Il progetto deve avere una wiki pubblica o privata a vostra scelta (**comunque su github**) che descrive obiettivi e contiene puntatori a:
  - API description document
  - Applicazione su Heroku
  - GitHub repository
  - La descrizione del progetto che voi realizzerete
  - La descrizione del processo SCRUM utilizzato
  - La wiki contiene inoltre i nomi dei partecipanti al progetto

Deadlines

- 23 ottobre:
  - GitHub repository creato e link inserito sul file di registrazione.
  - Condividetelo per cortesia con *renataguizzardi* and *jorgeramirez*
  - Ogni persona del gruppo deve fare una modifica a un file qualunque, ma uno stesso file (ad esempio un readme.md o readme.txt, dove ciascuno aggiunge il proprio nome e fa commit) e fare push sul github repo.
- 15 novembre:
  - *Product Backlog* creato in un *spreadsheet*
  - Stima e ordinazione delle *User stories*
  - Primo Sprint pianificato e *Sprint Backlog* creato

Deadlines

- 5 dicembre
  - Tutte le APIs implementate, *deployed* e *t*estate.

- 12 dicembre
  - Descrizione del processo SCRUM consegnata
  - Attività in classe di scambio di codice tra i gruppi.

- Nota:
  - a novembre, ci saranno altre *deadlines* per facilitare la esecuzione del progetto.

Grading

- 50% of the grade is the written exam
- 50% of the grade is the project (comprehending also an oral discussion about it)

- The students that do not attend class will only have a written exam. However, they will have to study a lot of extra material that was not covered in the classes.

Example

- Client: my uncle has a wine production
- We propose a System to help him manage his production
- User stories (Product backlog)

Implemented stories

  - The user may consult the system to make wine inventory, i.e. check which wines are in stock.
  - The user may insert/update information about new wine. Such information regards the grape, year, where it has been produced, how it tastes, smell, feel and look like, and what other things are or were produced in that land.
  - The user may delete information about a particular wine.
  - The user may see information about his wine in integration with market information regarding that kind of wine.
  - The user may verify how particular clients (which are important to him) like his wines.
  - The user may access a kind of lab, which helps him blend different kinds of wine to experiment new wine recipies.

Example

- Client: my uncle has a wine production
- We propose a System to help him manage his production
- User stories (Product backlog)

Example

  - The user may consult the system to make wine inventory, i.e. check which wines are in stock.
  - The user may insert/update information about new wine. Such information regards the grape, year, where it has been produced, how it tastes, smell, feel and look like, and what other things are or were produced in that land.
  - The user may delete information about a particular wine.
  - The user may see information about his wine in integration with market information regarding that kind of wine.
  - The user may access a kind of lab, which helps him blend different kinds of wine to experiment new wine recipies.
  - The user may verify how particular clients (which are important to him) like his wines.

Example of link to implementation

- **Description:** The user may consult the system to make wine inventory, i.e. check which wines are in stock.
- **Importance:** 100
- **Estimation:** 70
- **How to demo:** GET /products

Important links:

- https://swagger.io/docs/ (API documentation format)

- https://apiary.io/ (online editor for writing API docs)