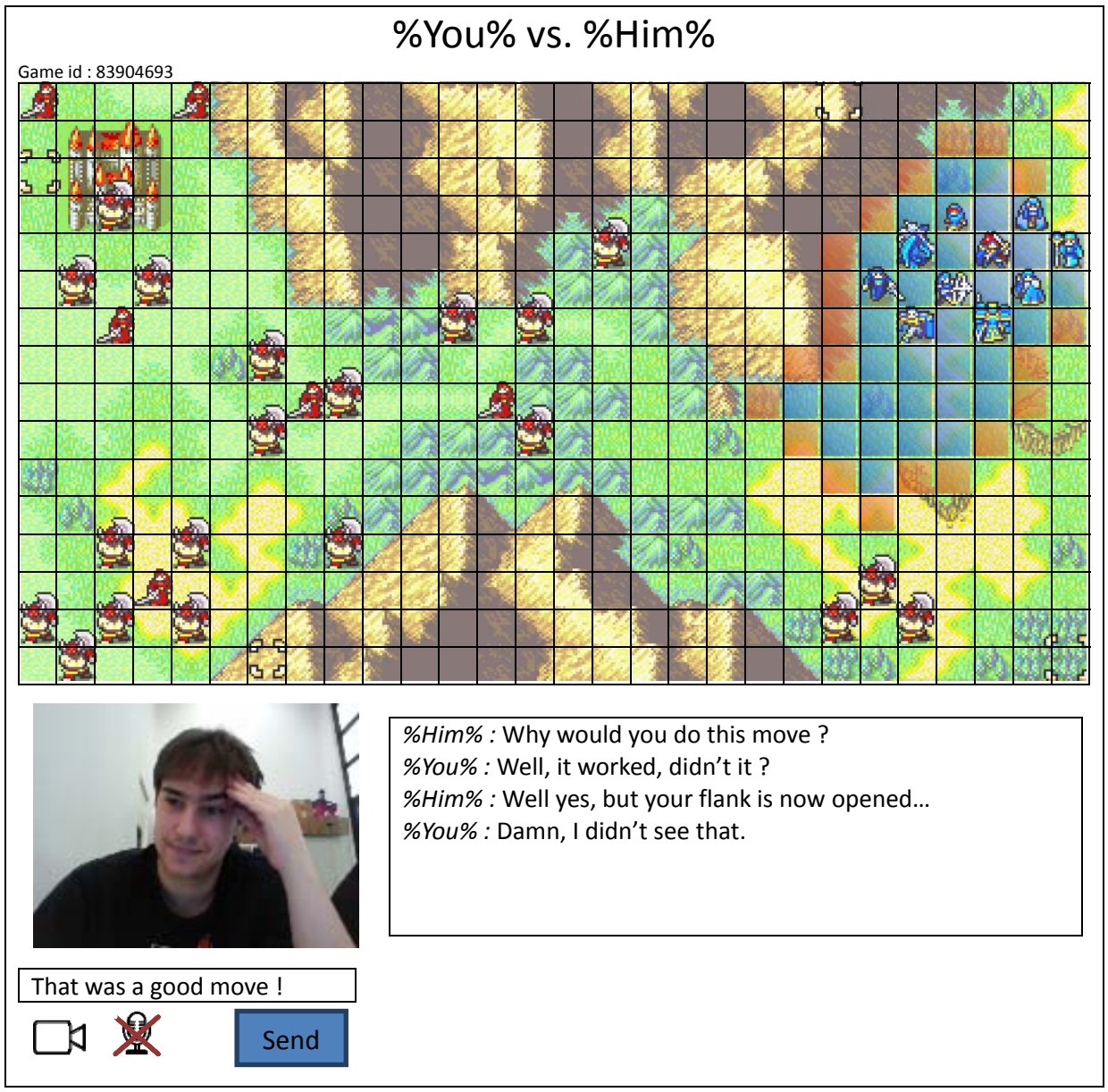
BACQUET Chloe - GAILLOT Antoine - KELLER Nicolas - MOUSSEAU Benjamin

Jeu de stratégie interactif sur internet utilisant les nouvelles technologies du web

Description

Créer une plateforme de jeu multijoueur ayant une interface qui permet aux utilisateurs de créer leur propre contenu, avec un système de visioconférence avec son adversaire pour recréer l'interaction humaine habituellement perdue dans un jeu multijoueur sur internet



Fonctionnalités :

- Système d'amis et de messagerie
- Bibliothèque de cartes avec éditeur
- Création de cartes par les utilisateurs
- Leaderboard, système de ranking
- Système de matchmaking automatisé
- Intégration aux réseaux sociaux
- Communication avec l'adversaire en visioconférence in-game via P2P

Technologies mises en place :

Pour travailler efficacement à plusieurs nous utiliserons le gestionnaire de version Git avec **GitHub**. Nous pourrons donc profiter du système de branchage pour permettre à chaque membre de travailler sur une fonctionnalité indépendante du projet et de passer facilement d'une branche à l'autre pour travailler à plusieurs sur un même problème.

Nous avons donc séparé le projet en plusieurs modules et sous-modules indépendants pour pouvoir travailler en parallèle. Coté serveur nous utiliserons le langage **PHP** coté serveur avec le framework **Symfony 2**. Dans le même esprit que **Git**, **Symfony** vas nous permettre de bien respecter le principe d'encapsulation et va nous permettre d'avoir un code structuré et consistant à travers tout le projet.

Nous utiliserons le moteur de template **Twig**. Ainsi nous marquons bien la séparation entre la vue et le modèle. Nous utiliserons le **DBAL Doctrine 2** basé sur **PDO**. Il nous permettra de faire abstraction de la couche MySQL et de se concentrer sur les fonctionnalités.

Coté client nous utiliserons les dernières technologies web, notamment le **HTML5** et les dernières API **JavaScripts** qui sont livrées avec directement dans le navigateur du client. Pour le design nous utiliserons le LESS que nous compilons en CSS3 à la volée sur le serveur avec la librairie **LESSPHP**. Nous aurons ainsi un code propre et modulable facilement maintenable que nous pouvons "minifier" en CSS3 pour optimiser les requêtes clients.

Coté JavaScript nous utiliserons le framework **jQuery** dans sa dernière version avec le CDN google par exemple. Toutes les fonctionnalités serrons codés sous forme de modules jQuery pour plus de clarté. jQuery sera particulièrement utile pour l'exploration du DOM et les échanges asynchrones avec le serveur.

Pour le jeu, nous utiliserons intensivement l'API **canvas** avec **WebGL** fournis par HTML5 pour gérer tout ce qui est rendu graphique. Nous utiliserons en outre la librairie JS **GameQuery** qui permet notamment de gérer simplement le temps et les layers de canvas.

Toujours pour le jeu, nous allons utiliser le protocole **WebRTC** qui permet de connecter deux machines distantes en P2P uniquement via le navigateur et l'API JS. Cette technologie va permettre d'alléger énormément la charge serveur et ainsi réduire les couts d'utilisations. Nous pourrons ainsi lancer des milliers de parties simultanées sur le même serveur alors qu'il n'aura pas pu supporter plus d'une 50aine de partie avec une architecture classique. De plus, cette technologie ouvre les porte de jeux qui aurait simplement été impossibles à réaliser à cause de la lenteur de l'architecture en triangle pour les jeux en temps réel. Nous pourrons faire passer les données du jeu, un flux vidéo et sonore et un chat par ce protocole.