

TFE

PLATEFORME WEB D'APPRENTISSAGE DE L'ANGLAIS PAR CHAT VIDEO

1. DESCRIPTION DU PROJET

CONCEPT DU PROJET

- ▶ Application web mettant en commun des personnes désireuses de pratiquer l'anglais
- ▶ Pratique de la langue via chat textuel et visioconférences directement intégrés à l'application
- ▶ Mise en communs de personnes ne parlant pas les mêmes langues excepté l'anglais, progression inévitable en utilisant l'anglais

CAHIER DE CHARGES

- ▶ Page d'accueil, enregistrement et connexion
- ▶ Menu de navigation, point central de l'utilisation de l'application
- ▶ Page de profil
- ▶ Système de gestion des amis
- ▶ Messagerie multi-conversations incluant la possibilité de visioconférence

2. MÉTHODOLOGIES DE DÉVELOPPEMENT

BEHAVIOUR DRIVEN DEVELOPMENT

- ▶ Apparenté au TDD
- ▶ Définitions des tests selon scénarios et user-stories :
 - ▶ As a - I want to - So That
- ▶ Ruby on Rails & RSpec
- ▶ Développement par itérations et suivant le cycle Red-Green-Refactor

- ▶ Détails techniques RSpec
 - ▶ Framework de testing BDD avec DSL
 - ▶ Expectations
 - ▶ Features, Requests
 - ▶ Capybara, ciblage du HTML
 - ▶ PhantomJS, navigateur de tests « fantôme »

ELEMENTS SUPPLÉMENTAIRES

- ▶ Git, Gitflow, GitHub
 - ▶ Branches, POCs, déploiement facilité
- ▶ Intégration continue via TravisCI
- ▶ Couverture du code CoderWall, codebase couverte à 100%

3. CONCEPTION DE L'APPLICATION

3. CONCEPTION DE L'APPLICATION

BASES DE L'APPLICATION

BASES DE L'APPLICATION



RUBY ON RAILS

- ▶ Ruby on Rails 4.2.5, framework développement web en Ruby, MVC, confort et rapidité de développement
- ▶ Système de gems, librairies tierces pouvant être incluses dans les projets Ruby

BASES DE L'APPLICATION



ACTIVE RECORD

- ▶ Conception de la base de données de manière itérative
- ▶ Object-Relationnal Mapping - ActiveRecord, abstraction du SGBD pour le développeur
- ▶ Système de migrations, utilisées pour définir les structure de données, historique des changements.

BASES DE L'APPLICATION

PREMIÈRES FEATURES

- ▶ Inscription, connexion, profil
- ▶ Design via Bootstrap et CSS3
- ▶ Système d'amitié, requêtes d'amitiés, association de polymorphisme ActiveRecord :
 - ▶ Appartenance d'un modèle plus d'un autre au travers d'une même association

3. CONCEPTION DE L'APPLICATION

IMPÉRATIFS TEMPS RÉEL

IMPÉRATIFS TEMPS RÉEL

PROBLÉMATIQUE

- ▶ Future implémentation du chat textuel et des visioconférences
- ▶ Rails seul n'est pas adapté au temps-réel
- ▶ Fonctionne de manière stateless
- ▶ L'application doit être statefull afin de pouvoir recevoir les nouveaux messages, notifications, à tout instant
- ▶ Possibilités avec Rails 5.0 ou ActionController:Live en gardant Rails

IMPÉRATIFS TEMPS RÉEL

SOLUTIONS EN RAILS PUR

- ▶ Rails 5.0
 - ▶ Utilise les websockets, mais encore en beta
- ▶ ActionController:Live
 - ▶ Etablit un canal de streaming de données sur la vue rendue
 - ▶ Utile pour de petites features
 - ▶ Peu de perspectives sur des besoins plus larges

IMPÉRATIFS TEMPS RÉEL

ANGULARJS, RAILS ET REST API

- ▶ Idée : utiliser Rails pour le backend uniquement et construire une single-page app en AngularJS
- ▶ Rails fournit une API RESTful qui va être utilisée par l'application Angular
- ▶ REST : Representational State Transfer, architecture web basée sur HTTP par Roy Fielding
Règles de conception, ressources identifiées par l'url, méthodes HTTP identifient les actions, etc...

IMPÉRATIFS TEMPS RÉEL

ANGULARJS, RAILS ET REST API

- ▶ REST ancré dans l'optique de Rails
- ▶ Angular étant en JavaScript, choix pertinent pour la mise en place d'une structure temps réel
- ▶ Besoin néanmoins de trouver une solution tierce pour faire communiquer Rails et Angular en temps réel

3. CONCEPTION DE L'APPLICATION

REFACTORING

REFACTORING

BACKEND RAILS

- ▶ Tests RSpec toujours valides pour le passage des vues en AngularJS
- ▶ Ecriture des tests de requêtes, conception de l'API en suivant la spécification JSON:API.
Les tests de requêtes fournissent aussi la documentation de l'API.
- ▶ Configuration de PhantomJS pour prendre en compte l'apparition du JavaScript dans l'application.
- ▶ Authentification via JSON Web Token, RFC7519, nécessité à terme d'utiliser HTTPS.

REFACTORING

FRONTEND

- ▶ Séparation de la gestion des librairies, composants frontend gérés par Bower, ceux du backend par le système de Gems
- ▶ Construction de la single-page app en Angular :
 - ▶ Routes, templates, contrôleurs et service
- ▶ Utilisation de l'API Rails via les services Angular

REFACTORING

BÉNÉFICES SUR L'APPLICATION

- ▶ Séparation complète des responsabilités des composants, ceux-ci sont plus ciblés et plus faciles à maintenir
- ▶ Logique plus claire et simple côté Rails
- ▶ Suppression du Ruby-embarqué, les anciennes vues Rails sont représentées en Angular sous un template HTML et un contrôleur JavaScript associé

3. CONCEPTION DE L'APPLICATION

CHAT TEXTUEL ET NOTIFICATIONS

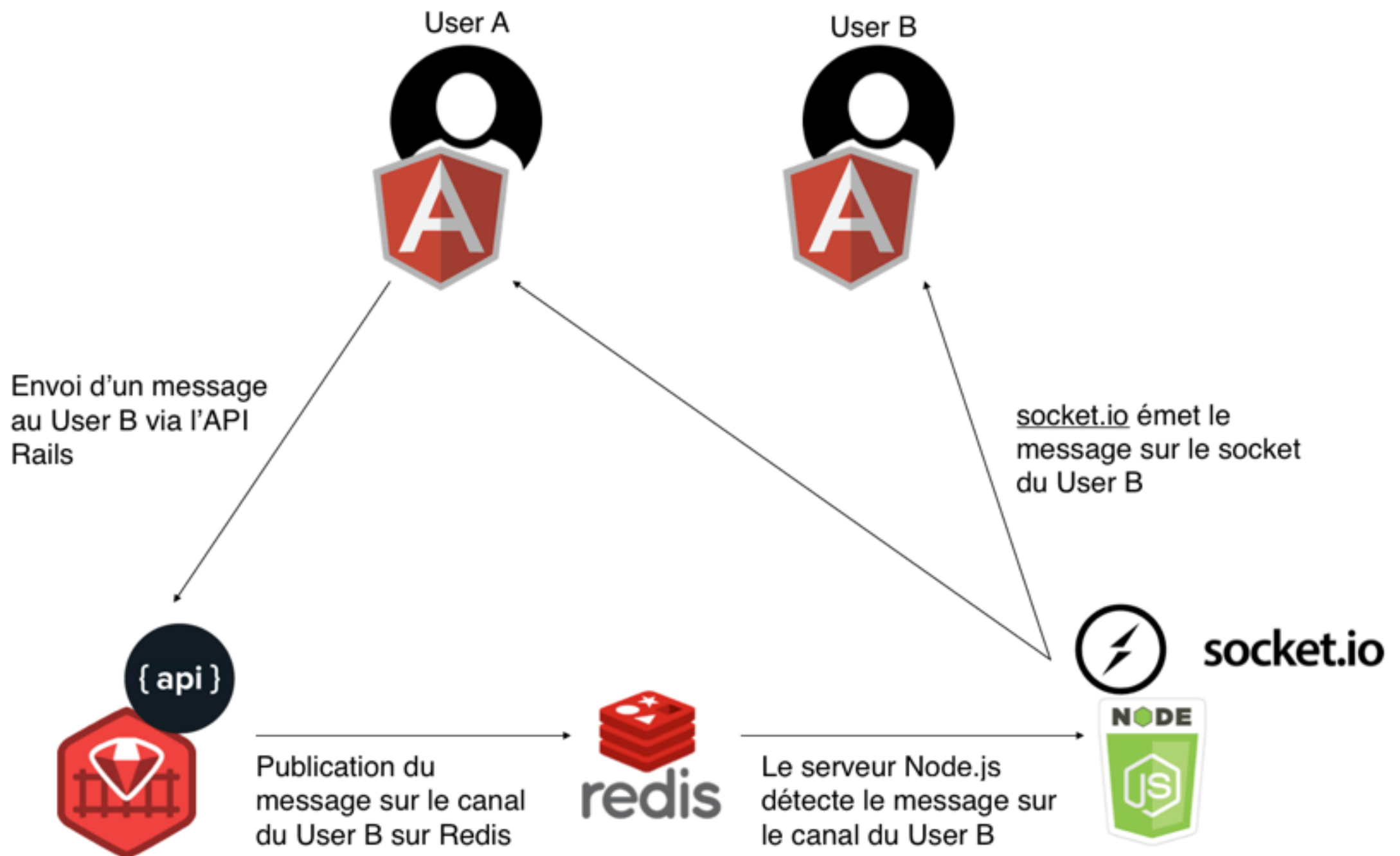
CHAT TEXTUEL ET NOTIFICATIONS

PROBLÉMATIQUE

- ▶ Comment mettre en relation Rails et Angular pour le temps réel? :
 - ▶ Redis: base de données NoSQL publish/subscribe
 - ▶ Socket.io : Similaire aux websockets, implémenté sur un serveur Node.js
- ▶ Rails et Angular communiqueront au travers d'une pile Redis et Socket.io

CHAT TEXTUEL ET NOTIFICATIONS

SCHÉMA DE FONCTIONNEMENT



3. CONCEPTION DE L'APPLICATION

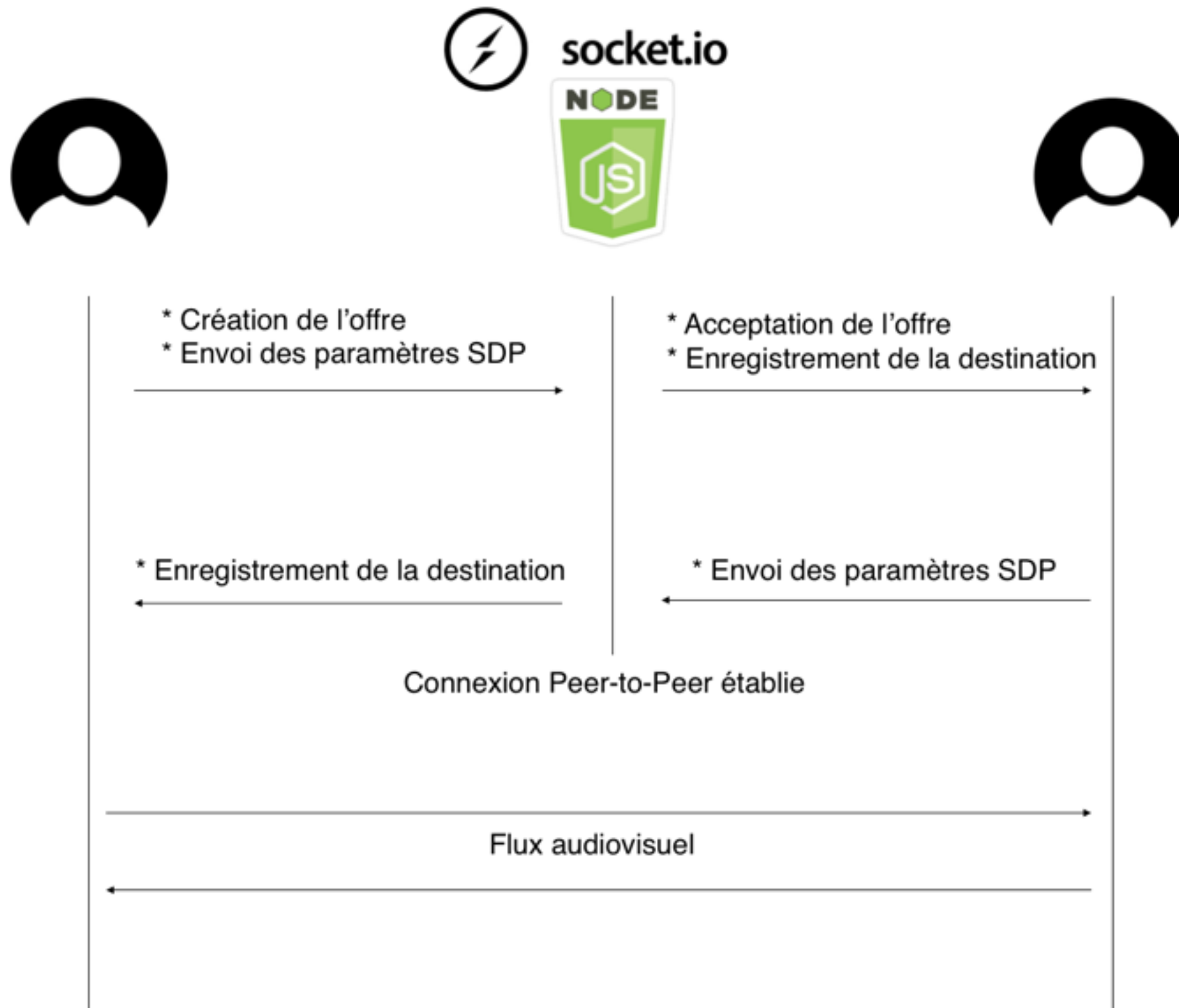
VISIOCONFÉRENCE

VISIOCONFÉRENCE

- ▶ Infrastructure temps-réel totalement adaptée
- ▶ WebRTC : API destinée aux communications en temps-réel
- ▶ Récupération du flux audiovisuel d'un utilisateur, ouverture de connexions peer-to-peer pour l'échange, etc...
- ▶ Utilisation HTTPS certifié par une CA obligatoire
- ▶ Architecture plus légère et indépendante de Rails

VISIOCONFÉRENCE

SCHÉMA DE FONCTIONNEMENT



4. CONCLUSION