**Project Name:** FITU

**Related Tech:** Mongodb, Nodejs, Angularjs

**Architecture Overview:**



**Environment setup:**

1. Install [mongodb](http://www.mongodb.org/), [nodejs](http://nodejs.org/).
2. Exec ‘npm install –g bower’, ‘npm install –g grunt’.
3. Get code from [git@gitrepoyanche.cloudapp.net:opt/git/project.git](mailto:git@gitrepoyanche.cloudapp.net:opt/git/project.git)
4. At root folder: exec ‘npm install’, ‘bower install’ to resolve dependencies.
5. Go folder: fitu/documents, run commands under ‘db clean up.txt’ in mongo shell, to initialize local db.
6. At root folder: exec ‘grunt dev’, to build the project, one more folder: statics, should appear under the root folder.
7. At root folder: exec ‘grunt watchuser’ if you’re developing user client, ‘grunt watchvendor’ if you’re developing vendor client, ‘grunt watchadmin’ if you’re developing admin client.
8. Local dns, hosts file in Windows:

127.0.0.1 www.fitu.com

127.0.0.1 admin.fitu.com

127.0.0.1 api.fitu.com

127.0.0.1 vendor.fitu.com

1. Exec ‘mongod’ to start mongo
2. ‘cd fitu’, Exec ‘node start.js dev’ on fitu sub folder to start local debug
3. PS. If you want to generate prod version files, use ‘grunt prod’ or ‘grunt test’ for test env.

**Data Contract (http):**

**Request**

GET: read

POST: create

PUT: update

DELETE: delete

**Response**

200 🡪 success

400 🡪 argument issue

401 🡪 no authorization

405 🡪 functionality not allowed

409 🡪 conflict, email/nick name duplication

500 🡪 server side error

**Resource List**

1. **Activities**

**http://api.fitu.com/activities?id&siteId&siteIdleId&vendorId&page&pageSize**

**GET:**

**Available filters:** id siteId siteIdleId vendorId page pageSize

**Filter priority:** id > siteId > siteIdleId > vendorId, exclusive

**Auth:** no auth required

**Response:** application/json

id: 32 bits id,

site: {id:, name:, location: {address: string, geo: string}, picUrl: },

siteIdle: {id:, startsOn:, endsOn: js date string, slots: number },

creator: {id:, headUrl, personal: },

name: string,

startsOn: js date string,

endsOn: js date string,

createdOn: js date string,

createdBy: 32 bits id,

tags: [] array of string,

intro: string,

picUrl: image url,

capacity: number

**POST:**

**Available Context:** siteId(required) siteIdleId(optional, must belongs to the site indicated by siteId)

**Auth:** Login required

**Request Format:**

name,

startsOn:,

endsOn:,

tags:,

intro:,

picUrl:,

capacity:

**Response:**

Success: text/plain, headers: Location: http://api.fitu.com/activities?id=new created id

Failed: http code(400,401…) TODO: specify bad field.

**PUT:**

**Available Context:** id

**Auth:** need to be the activity creator

**Request Format:**

name,

tags:,

intro:,

picUrl:,

capacity:

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

**DELETE:**

**Available Context:** id

**Auth:** need to be god

**Response:** text/plain, nothing

1. **Vendors**

**http://api.fitu.com/vendors?id&page&pageSize**

**GET:**

**Available filters:** id page pageSize

**Auth:** no auth required

**Response:** application/json

id: 32 bits id,

tags: [],

location: {address:, geo: },

intro: string,

logoUrl:,

owner: {id:, headUrl:, personal: },

admins: [ {id:, headUrl:, personal: } ],

**POST:**

**Available Context:**

**Auth:** God required

**Request Format:**

tags: [],

location: {address:, geo: },

intro: string,

logoUrl:,

ownerId:,

admins: [ array of 32 bit id ],

Success: text/plain, headers: Location: http://api.fitu.com/vendors?id=new created id

Failed: http code(400,401…) TODO: specify bad field.

**PUT:**

**Available Context:** id

**Auth:** owner or admin

**Request Format:**

tags: [],

location: {address:, geo: },

intro: string,

logoUrl:,

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

**DELETE:**

**Available Context:** id

**Auth:** need to be god

**Response:** text/plain, nothing

1. **Sites**

**http://api.fitu.com/sites?id&vendorId&page&pageSize**

**GET:**

**Available Filter:** id vendorId page pageSize

**Filter priority:** id > vendorId, exclusive

**Auth:** no auth required

**Response:**

id:,

name:,

tags: [],

location:{ address:, geo:, },

intro:,

picUrl:,

vendor: {id:, name:, logoUrl: }

**POST:**

**Available Context:** vendorId (required)

**Auth:** Vendor admin/owner

**Request Format:**

name:,

tags: [],

location:{ address:, geo:, },

intro:,

picUrl:,

**Response:**

Success: text/plain, headers: Location: http://api.fitu.com/sites?id=new created id

Failed: http code(400,401…) TODO: specify bad field.

**PUT:**

**Available Context:** id

**Auth:** vendor owner or admin

**Request Format:**

name:,

tags: [],

location:{ address:, geo:, },

intro:,

picUrl:,

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

**DELETE:**

**Available Context:** id

**Auth:** need to be god

**Response:** text/plain, nothing

1. **SiteIdles**

**http://api.fitu.com/siteIdles?id&vendorId&siteId&page&pageSize**

**GET:**

**Available Filter:** id siteId vendorId page pageSize

**Filter priority:** id > siteId > vendorId, exclusive

**Auth:** no auth required

**Response:**

id:,

slots: number,

price: number,

startsOn:,

endsOn:

site: {id:, name:, location: {address, geo: }, picUrl: },

createdOn:

**POST:**

**Available Context:** siteId

**Auth:** Vendor admin/owner required

**Request Format:**

slots: number,

price: number,

startsOn:,

endsOn:

**Response:**

Success: text/plain, headers: Location: http://api.fitu.com/siteIdles?id=new created id

Failed: http code(400,401…) TODO: specify bad field.

**PUT:**

**Available Context:** id

**Auth:** Vendor admin/owner required

**Request Format:**

slots: number,

price: number,

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

**DELETE:**

**Available Context:** id

**Auth:** need to be god

**Response:** text/plain, nothing

1. **Members**

**http://api.fitu.com/members?id&userId&actId&page&pageSize**

**GET:**

**Available Filter:** id userId actId (at least one) page pageSize

**Filter priority:** id > userId, actId (same level)

**Auth:** actId: activity creator, userId: operator himself, id: member of organizer or member of operator self

**Response:**

id:,

user: {id:, personal: {nickName: string, phone: phone string, contact: email string, gender: ‘男’ or ‘女’ or ‘’ }, headUrl: image url },

activity: {id:, name:, startsOn:, endsOn:, picUrl: },

createdOn:,

statusId: 10 queued, 20 confirmed, 30 quit

**POST:**

**Available Context:** actId

**Auth:** login required

**Request Format:** nothing

**Response:**

Success: text/plain, headers: Location: http://api.fitu.com/members?id=new created id

Failed: http code(400,401…) TODO: specify bad field.

**PUT:**

**Available Context:** id

**Auth:** self, act organizer

**Request Format:**

statusId: user -> quit, act organizer -> confirmed, queued

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

**DELET:**

**Available Context:** id

**Auth:** need to be god

**Response:** text/plain, nothing

1. **Users**

**http://api.fitu.com/users?id&page&pageSize**

**GET:**

**Available Filter:** id page pageSize

**Auth:** self(id), god(all)

**Response:**

id:,

personal: {nickName: string, phone: phone string, contact: email string, gender: ‘男’ or ‘女’ or ‘’ },

headUrl:,

email:,

**POST:**

**Auth:** no auth required

**Request Format:**

personal: {nickName: string, phone: phone string, contact: email string, gender: ‘男’ or ‘女’ or ‘’ },

headUrl:,

email:,

hash\_pwd:

**Response:**

Success: text/plain, headers: Location: http://api.fitu.com/users?id=new created id

Failed: http code(400,401…) TODO: specify bad field.

**PUT:**

**Available Context:** id

**Auth:** self

**Request Format:**

personal: {nickName: string, phone: phone string, contact: email string, gender: ‘男’ or ‘女’ or ‘’ },

headUrl:,

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

**DELET:**

**Available Context:** id

**Auth:** need to be god

**Response:** text/plain, nothing

1. **UserContacts**

**http://api.fitu.com/usercontacts?query**

**GET:**

**Available Filter:** query (by nickName or contact email)

**Auth:** owner/admin of vendor

**Response:**

id:,

personal: {nickName: string, phone: phone string, contact: email string, gender: ‘男’ or ‘女’ or ‘’ },

headUrl:,

1. **UserCredentials**

**http://api.fitu.com/usercredentials?userId**

**PUT:**

**Available Context:** userId

**Auth:** self

**Request Format:**

email: (change user email to this)

hash\_pwd: (change user pwd to this)

confirm\_hash\_pwd: (current password)

**Response:**

Success: text/plain, nothing

Failed: http code(400,401…) TODO: specify bad field.

1. **Sessions**

**POST:**

**Auth:** no auth required

**Request Format:**

email:,

hash\_pwd:

**Response:**

Success: application/json

sessionId:

userId:

Failed: http code(400,401…) TODO: specify bad field.